EIM Services
DP- 2/3
Atlas & Thor3

Checkout and Launch Control System (CLCS)

84K00520-570

Approval:

Chief, System Engineering and Integration Division Date

Date

Date

Date

Date

Date

Prepared By:

Date

NOTE: See “**Supporting Document Note**” on following page

REVISION HISTORY

REV	DESCRIPTION	DATE
BASIC		

Supporting Document Note:

Acronyms and definitions of many common CLCS terms may be found in the following documents: CLCS Acronyms 84K00240 and CLCS Project Glossary 84K00250.

Table of Contents

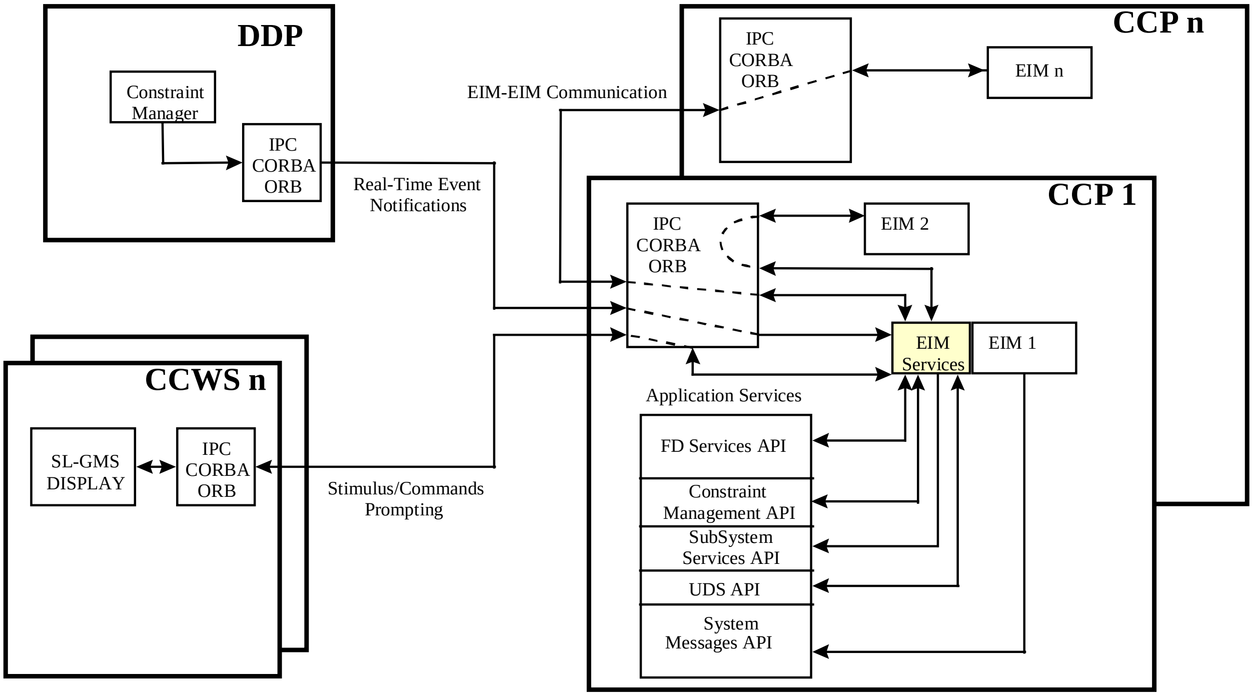
1. EIM SERVICES

1.1 EIM Services Introduction

1.1.1 EIM Services Overview

The EIM Services CSC provides wrappers for Applications SW to communicate with other Application Services on the RTPS. The wrappers provide an interface to FD Services, constraint management, user prompting, sub-system services, and other systems and services required by the Applications SW. EIM Services reside on the CCP within the scope of the Applications SW using ControlShell ™ application development CASE tool. (EIM Services' components will be linked into the application – not a separate process.) EIM Services provide ControlShell ™ components which bring the Application Services into the visual framework of ControlShell ™. This allows the Applications SW designers to concentrate on the systems design and not the application services details. A repository of components will provide the required RTPS functions needed by Applications SW.

End Item Management Services is a collection of components that encapsulate Application Service's APIs providing an interface between User Applications and Application Services. This protects applications from any changes in the underlying Application Services interface, as well as protecting application services from changes in user application tools.



1.1.2 EIM Services Operational Description

The EIM Services CSC shall provide components for the Applications SW to communicate with Application Services on the CCP. Commands originating in the CCWS will interface with the Applications SW through CORBA. The Applications SW will provide control and monitoring with Finite State Machines (FSMs) and Composite Object Groups (COGs). COGS and FSMs will contain EIM Service's components for interfacing with Application Services. EIM Services shall call the API's to update FD values, send FD commands, set Constraints, display dialog windows, and issue commands to gateways. EIM Services shall use CORBA to receive and send stimuli/commands from other applications, receive Constraint Notifications, and for other inter-process communication.

1.2 EIM Services Specifications

1.2.1 EIM Services Groundrules

- The components are only a thin wrapper interface and will simply pass on any Application Service's return codes, errors back to the calling End Item Manager application. (Exceptions are caught & system messages are sent.)
- EIM Services makes no distinction between pseudo FD's, Fused FD's and real FD's.
- EIM Services code will be posix compliant.
- EIM Services expects to be provided:
  - FD Services: API
  - EIM App Software: Requirements, Priorities
  - Constraint Management Services: API
  - Sub-system Services: API
  - Logging Services: API
  - ControlShell ™: Licenses, Training, Full Release with menu typing capability
- The Test EIM application will be implemented by EIM Services.
- ERP Issue 128: Local Logging (per SLS 2.2.10.1.17) is not required by Applications SW, and never envisioned to be needed since:
  - Debugging will be done via system messages
  - Checkpoint/Restart handles local file storage/access
  - Configuration snapshots are handled in the SDC

1.2.2 EIM Services Functional Requirements

This section contains a list of SLS and high level derived requirements that are driving the design.

NOTE: ‡ denotes functionality delivered in Thor3

(SLS 2.2.3.3.3) The CLCS shall provide the capability to restrict issuance of commands to only authorized users and applications.

(SLS 2.2.4.1.1) The CLCS shall provide, using a Graphical User Interface (GUI) paradigm, the capabilities identified in Appendix C with a Y in the column titled Implement.

(SLS 2.2.5.1.1) The RTPS shall provide fault tolerant End-Item monitoring and control.

(SLS 2.2.5.1.2) The RTPS shall provide End-Item monitoring and control definition that is directly understandable in form and content by End-Item engineering personnel.

(SLS 2.2.5.1.3) The RTPS shall provide a layered technique for defining End-Item monitoring and control that supports reuse and reduction of time to implement.

(SLS 2.2.5.1.5) The RTPS shall provide the capability to monitor all End-Item FD's current state against expected state during all operational End-Item phases.

(SLS 2.2.5.1.~~6~~5) The CLCS shall provide effective support to test director and test management personnel to clearly understand End-Item and test summary status.

(SLS 2.2.5.1.7) The CLCS shall provide effective support to test director and test management personnel to efficiently and reliably control and track testing progress.

Constraint Management:

(SLS 2.2.5.4.~~2~~1) CLCS shall provide the capability for multiple (TBD number) users and system or user applications to request notification of constraint events for each Measurement FD.

(SLS 2.2.5.4.~~3~~2) CLCS shall provide the capability for each user, and system or user application requesting constraint notification to specify the limits/condition under which they will be notified.

End-Item Management:

The End-Item Management function provides the capability for users to create End-Item Manager applications to perform closed loop control of a specific End-Item system or component.

(SLS 2.2.5.6.3) RTPS shall provide the capability to delegate continuous End-Item monitoring to Constraint Management and respond to Constraint Management notification events.

(SLS 2.2.5.6.4) RTPS shall provide the capability to operate an End-Item using rate based control.

(SLS 2.2.5.6.6) RTPS shall provide the capability to alter the state of frame rate/time domain control based on an event.

(SLS 2.2.5.6.8) RTPS shall provide an End-Item control layer/encapsulation allowing maintenance in a single place and reuse through out user applications.

This section contains derived requirements.

1.2.2.1 ‡ The API shall provide a method to issue values to analog output FD’s to support on-board port/MDM specifications.
1.2.2.2 The capability shall be provided to support on-board/MDM specifications in discrete commands.
1.2.2.3 The API shall provide a method for determining which system asserted a constraint when an FD is marked with a constraint limit violation.
1.2.2.4 The API shall provide a method for changing the constraint condition for an enumerated FD.
1.2.2.5 ‡ The API shall provide a method for changing the change processing value of an analog FD.
1.2.2.6 ‡ The API shall provide a method for changing the data health for any FD.
1.2.2.7 ‡ The API shall provide a method for changing calibration coefficients.
1.2.2.8 ‡ The API shall provide a method to activate or inhibit stale data checking on a per FD basis.
1.2.2.9 ‡ The API shall provide a method for reading the current change processing value for an analog FD.
1.2.2.10 ‡ The API shall provide a method to read the current data health for any FD.
1.2.2.11 ‡ The API shall provide a method to activate or inhibit stale data checking on any gateway.
1.2.2.12 The API shall provide a method to activate or inhibit data fusion limit checking for any class of limits.
1.2.2.13 Functionality shall be provided to control the SRB MDM’s (lock/unlock). (Reference KSC-LPS-OP-033-4 Section 6.0)
1.2.2.14 The API shall provide a method to set CDT Hold
1.2.2.15 The API shall provide a method to Change Data Health on a Group FD (DD)
1.2.2.16 The API shall provide a method for Change Display Attributes (DD
1.2.2.17 The API shall provide a method for Present Value of (DD
1.2.2.18 The API shall provide a method for MDM Read Via prom (Power Up/Down on LDB GW)
1.2.2.19 The API shall provide a method to Switch Bus (Power Up/Down on LDB GW)
1.2.2.20 The API shall provide a method to Switch Mode (Power Up/Down on LDB GW)
1.2.2.21 The API shall provide a method to Resume (Power Up/Down on LDB GW)
1.2.2.22 The API shall provide a method to Read SIO Devices (Power Up/Down on LDB GW)
1.2.2.23 The API shall provide a method to Read MEC Flight (Power Up/Down on LDB GW)
1.2.2.24 The API shall provide a method to Read MEC CAP (Power Up/Down on LDB GW)
1.2.2.25 The API shall provide a method to Read MEC PIC (Power Up/Down on LDB GW)
1.2.2.26 The API shall provide a method to Read MEC Preflight (Power Up/Down on LDB GW)
1.2.2.27 The API shall provide a method to Command MEC Master (Power Up/Down on LDB GW)
1.2.2.28 The API shall provide a method to Command MEC Wrap (Power Up/Down on LDB GW)
1.2.2.29 The API shall provide a method to Command MEC SPCV (Power Up/Down on LDB GW)
1.2.2.30 The API shall provide a method for reading the current value of an analog FD in raw counts.
1.2.2.31 The API shall provide a method for reading the current value of a discrete FD in unprocessed format.
1.2.2.32 The API shall provide a method to set the systems CDT/MET.
1.2.2.33 The API shall support one-to-one and one-to-many options for the issuance of values to FD's
1.2.2.34 The API shall provide a method for reading the current value of a discrete FD in engineering units.

1.2.2.35 The API shall provide a method to read the time of the last change in value of the FD.

1.2.2.36 The API shall provide a method for issuing a value to digital pattern output FD’s.

1.2.2.37 The API shall validate that issued values are compatible with Function Designator types prior to issuing the command.

1.2.2.38 The API shall be able to pass engineering unit data types across interface boundaries in a type safe way.

1.2.2.39 The API shall provide an efficient type safe mechanism for manipulating computations involving engineering units.

1.2.2.40 The API shall provide a method for setting a discrete output FD’s.

1.2.2.41 The API shall provide a method for reading the current value of an analog FD in engineering units.

1.2.2.42 The API shall provide a method to read the health status the last change of an FD.

1.2.2.43 The API shall provide a method to identify function designators that shall be delivered via queued service, providing access to every change value in time ordered fashion.

1.2.2.44 The API shall provide a method to read the next value of a multi-sample queued function designators.

1.2.2.45 The API shall provide a method to read the next N values of a multi-sample queued function designator.

1.2.2.46 The API shall provide a method to clear all queued samples pending for the application.

1.2.2.47 The API shall provide a method to cancel queued function designator delivery by FD.

1.2.2.48 The API shall provide a method for querying the on-line data bank by FDID or FD name.

1.2.2.49 The API shall provide a method for querying any piece of information stored in the on-line data bank for a particular FD.

1.2.2.50 The API shall provide a method to issue a prompt to, and receive response from, a user display window.

1.2.2.51 The API shall preserve the current LPS engineering unit model. The API shall be able to work with temperatures, pressures, discrete states, and enumerated states.

1.2.2.52 (The API shall provide a method to translate API error codes into system text messages which may be displayed for the user.

1.2.2.53 The API shall provide a method for reading the return to limits indicator (event) of an digital pattern / Enumerated FD.

1.2.2.54 The API shall provide a method for reading an FD's high limit constraint indicator (event) for all analog constraint limit sets.

1.2.2.55 The API shall provide a method to apply values to analog output FD’s.

1.2.2.56 The API shall set discrete output FD’s using the literal key words OPEN, CLOSE, TRUE, FALSE, WET, DRY,ON, OFF.

1.2.2.57 The API shall provide a method to specify a time value for a discrete command. This command shall set the command to the indicated state for the specified period and then return it to the original state.

1.2.2.58 The API shall provide a method for issuing values to a pseudo function designator.

1.2.2.59 The API shall provide a method for changing the constraint limits associated with an analog FD.

1.2.2.60 The API shall provide a method for changing the constraint state of a discrete FD.

1.2.2.61 The API shall provide a method for changing the constraint condition for a digital pattern FD.

1.2.2.62 The API shall provide a method to activate or inhibit constraint checking associated with an FD for an application.

1.2.2.63 The API shall provide a method for reading the FD constraint indicator (event) for an FD.

1.2.2.64 The API shall provide a method for reading the return to limits indicator (event) of an FD.

1.2.2.65 The API shall provide a method for reading an FD's low limit indicator (event) for all analog constraint limit sets.

1.2.2.66 The API shall provide methods for communicating between concurrently executing applications as described in 4.5.1.3 above.

1.2.2.67 The API shall provide a method for reading the current value of a digital pattern FD and enumerated type FD.

1.2.2.68 The API shall provide a method for determining if a function designator's health status is OK, FAILED, or WARNING.

1.2.2.69 The API shall provide a method of reading the detailed health status from the health status word for a function designator. The detailed health shall identify the following conditions:

1.2.2.70 The API shall provide a method for determining if processing active or inhibited for this FD.

1.2.2.71 The API shall provide a method for determining if application advisory notification active or inhibited for this FD.

1.2.2.72 The API shall provide a method for determining if engineering bypass active or inhibited for this FD.

1.2.2.73 The API shall provide a method to change an FD’s status indicator (event) in order to mark the measurement bad or good.

1.2.2.74 The API shall provide a method to specify and cancel event notification and an event handler for the occurrence of an FD constraint violation.

1.2.2.75 The API shall provide a method to activate or inhibit all FD constraint notifications active for the application.

1.2.2.76 The API call shall return an indicator to the calling process which indicates success or failure of an API call and the reason associated with any failure condition.

1.2.2.77 The API shall provide a method of indicating why a command has failed.

1.2.2.78 The API shall provide a method for reading the constraint limits associated with an analog FD for an application.

1.2.2.79 The API shall provide a method for changing the sample rate of any GSE FD.

1.2.2.80 The API shall provide a method for reading the constraint conditions associated with a discrete FD for an application.

1.2.2.81 The API shall provide a method of reading the current sample rate of an FD.

1.2.2.82 The API shall provide a method to read the constraint limit processing status in the data fusion function.

1.2.2.83 The API shall support strong compile time checking and external symbol resolution. The API shall minimize the need for run-time dependency checking.

1.2.2.84 ‡ The API shall provide a method to activate or inhibit measurement processing on a per FD basis.

1.2.2.85 The API shall provide the capability to read the current hardware address of an FD.

1.2.2.86 The API shall provide a method for reading the current data stale indicator for an FD.

1.2.2.87 The API shall provide a method to specify application error routines which are called on the occurrence of user specified error conditions.

1.2.2.88 EIM Services will provide ControlShell ™ repositories:

- A repository (Configuration controlled).

1.2.2.89 EIM Services will provide timing functionality:

- *Stopwatch*

FD Services:

1.2.2.90 EIM Services will provide ControlShell ™ component wrappers to Application service's FD Services API's for each FD data type:

- Analog measurement
- Analog stimulus
- Discrete measurement
- Discrete stimulus
- Digital Pattern measurement
- Digital Pattern stimulus
- Pseudo Analog
- Pseudo Discrete
- Pseudo Digital Pattern
- ‡ Enumerations
- ‡ Pseudo String

Constraint Management:

1.2.2.91 EIM Services shall provide a capability for Applications SW to Assert, Alter, Activate, Inhibit, and Release Atomic Constraints.

1.2.2.92 EIM Services shall provide a capability for Applications SW to Assert, Alter, Activate, Inhibit, and Release Compound and Summary Constraint.

1.2.2.93 EIM Services will provide a capability to Alter, Activate, and Inhibit RCL Constraints by FDName.

1.2.2.94 The EIM Interface to the Constraint Management API will be through EIS Constraint STCs and ATCs.

1.2.2.95 EIM Services shall provide a capability to receive and react to Real-time Constraint Notifications from Constraint Management.

1.2.2.96 Generic Event Handling ATC components will be provided to handle Constraint Notifications and send Stimuli to EIM FSMs.

Inter-Process Communications:

1.2.2.97 ‡ EIM Services shall provide ControlShell ™ components for EIM construction, allowing EIM's to interact with other processes, including other Applications SW, implemented via Common Object Request Broker Architecture (CORBA).

1.2.2.98 ‡ EIM Services shall provide a component pair that allows an End Item Manager to change the state within another EIM's Finite State Machine (FSM).

1.2.2.99 ‡ EIM Services shall provide a series of component pairs that allow Applications SW to use the method pins of components in other Applications SW:

- *NoArg.cm user and provider components.*
- *EIS_CSTypeDouble_cm.cm user and provider components.*
- *EIS_CSTypeInt_cm.cm user and provider components.*
- *EIS_DigitalPattern_cm.cm user and provider components.*
- *EIS_Analog_cm.cm user and provider components.*
- *EIS_Discrete_cm.cm user and provider components.*

User Display Services:

1.2.2.100 EIM Services shall provide ControlShell ™ component wrappers to the Application service's User Display Services API's:

- *Message Dialog (One Button Dialog)*
- *Two Button Dialog*
- *Three Button Dialog*
- *Four Button Dialog*
- *‡ String Input Dialog*
- *‡ Floating Point Numeric Input Dialog*
- *‡ Integer Numeric Input Dialog*

1.2.2.101 EIM Services shall use CORBA for inter-process communications between the component wrappers and the UDS Dialogs.

1.2.2.102 EIM Services shall provide a mechanism for Applications SW to recover in case of a CORBA call failure or if a specified amount of time passes before it receives a user response from a Dialog.

Sub-system Services:

1.2.2.103 ‡ EIM Services shall provide the capability for Applications SW to inhibit data acquisition on a gateway.

1.2.2.104 ‡ EIM Services shall provide the capability for Applications SW to activate data acquisition on a GSE gateway.

1.2.2.105 ‡ EIM Services shall provide the capability for Applications SW to activate data acquisition on a LDB gateway.

1.2.2.106 ‡ EIM Services shall provide the capability for Applications SW to activate or inhibit data processing on a gateway.

1.2.2.107 ‡ EIM Services shall provide the capability for Applications SW to activate or inhibit change processing on a gateway.

1.2.2.108 ‡ EIM Services shall provide the capability for Applications SW to activate or inhibit command issuance on a GSE gateway.

1.2.2.109 ‡ EIM Services shall provide the capability for Applications SW to activate or inhibit command issuance on a LDB gateway.

1.2.2.110 ‡ EIM Services shall provide the capability for Applications SW to activate or inhibit HIM polling on a GSE gateway.

1.2.2.111 ‡ EIM Services shall provide the capability for Applications SW to activate or inhibit HIM testing on a GSE gateway.

1.2.2.112 ‡ EIM Services shall provide the capability to activate or inhibit frame logging on a PCM gateway.

1.2.2.113 ‡ EIM Services shall provide the capability to change the selected PCM downlink gateway.

1.2.2.114 ‡ EIM Services shall provide the capability to change the synch bits on a PCM gateway.

1.2.2.115 ‡ EIM Services shall provide the capability to activate or inhibit data acquisition on a PCM gateway.

1.2.2.116 ‡ EIM Services shall provide the capability to activate or inhibit command issuance on a CS gateway.

1.2.2.117 ‡ EIM Services shall provide the capability to activate or inhibit HIM polling on a CS gateway.

1.2.2.118 ‡ EIM Services shall provide the capability to activate or inhibit HIM testing on a CS gateway.

1.2.2.1 EIM Services Invocation and Initialization Requirements

No Unique Initialization Requirements for Atlas

1.2.2.2 EIM Services Error Handling and Error Conditions

1.2.2.2.1 FD Services Interfaces Error Conditions

Error Condition	Known Causes	Response
FDNotFoundException	FDDictionary lookup() failed	Record system message #296 Return –1, stop ControlShell ™
FDTypesDontMatchException	FDDictionary lookup() failed	Record system message #397 Return –1, stop ControlShell ™
FDWriteException	FD services will throw this exception on failed set(), issue(), or apply() methods.	Record system message #391 Return –1 to calling function.
FDReadException	FD services will throw this exception on failed update() or value() methods.	Record system message #392 Return –1 to calling function.
FDHealthWriteException	FD services will throw this exception on failed set(), issue(), or apply() methods.	Record system message #393 Return –1 to calling function.

1.2.2.2.2 User Display Services Interfaces Error Conditions

Error Condition	*Known Causes*	*Response*
CORBA::Exception	*UDS Server is not running on CCWS.*	*Record system message* *ASV_EIS_CORBA_EXCEPTION* *Return –1, stop ControlShell ™*
pthread_create() returns an error	*Pthread library was unable to create a new thread*	*Record system message* *ASV_EIS_PTHREAD_FAILED* *Return –1, stop ControlShell ™*

1.2.2.3 EIM Services Unique Algorithm Requirements

N/A

1.2.2.4 EIM Services Performance Instrumentation

N/A

1.2.2.5 Checkpointed Data Requirements

N/A

1.2.3 EIM Services Performance Requirements

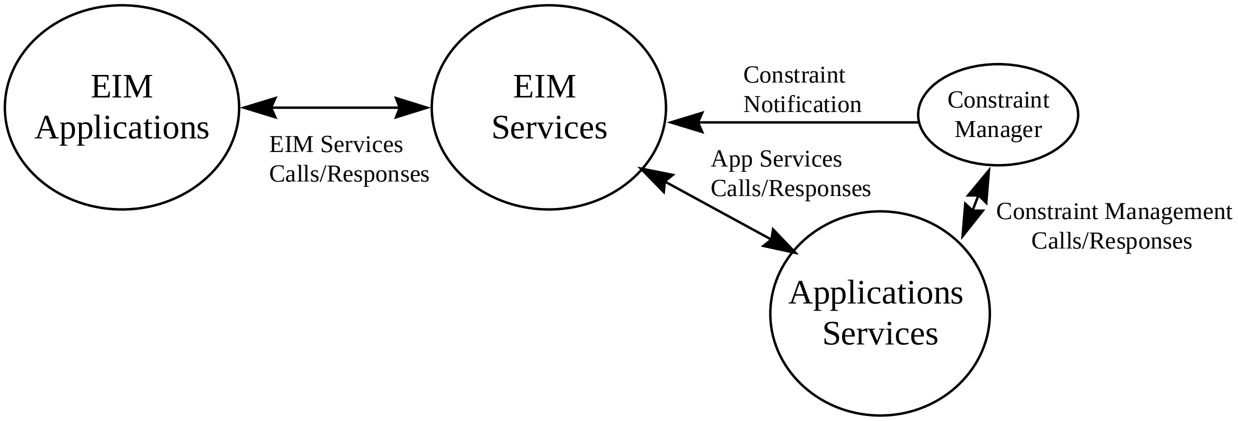
None determined for Atlas

1.2.4 Performance Allocation Budget

N/A

1.2.5 EIM Services Interfaces Data Flow Diagrams

This diagram provides a pictorial representation of the data flow between EIM Services, EIM Applications and Application Services.


EIM Applications
EIM Services Calls/Responses
EIM Services
Constraint Notification
Constraint Manager
App Services Calls/Responses
Constraint Management Calls/Responses
Applications Services

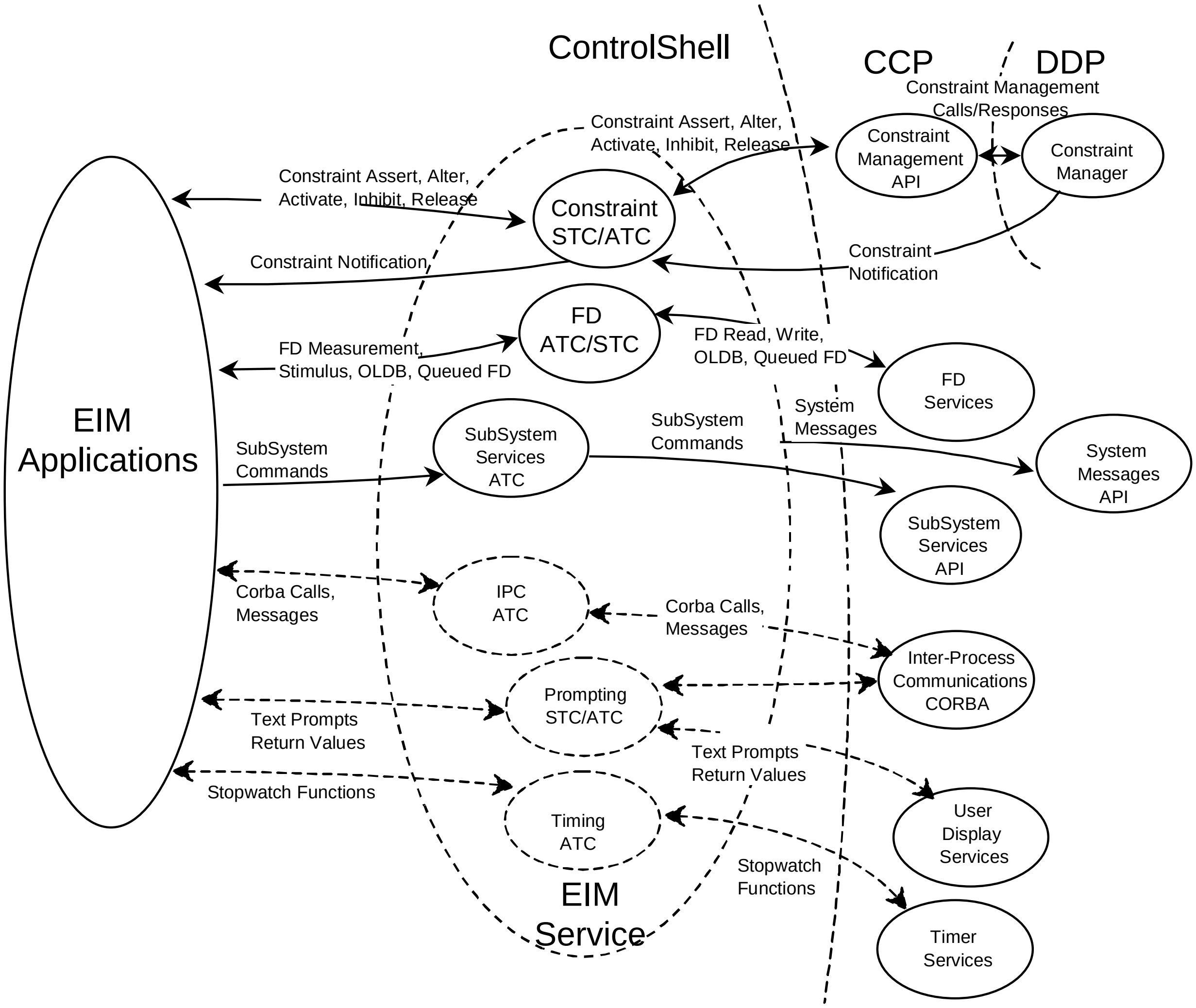
Note: **This is the end of the Design Panel 2 Required material. The information covered below is for Design Panel 3.**

1.3 EIM Services Design Specification

1.3.1 EIM Services Detailed Data Flow

This diagram provides a pictorial representation of the data flow between EIM Services, Applications SW and Application Services objects.

Detailed Data Flow Diagram



1.3.2 EIM Services System Context Diagram

See 1.3.1 and 1.2.5

1.3.3 EIM Services State Definition and State Transition Diagram

N/A

1.3.4 EIM Services Unique Algorithm Design

N/A

1.3.5 EIM Services Development Tools

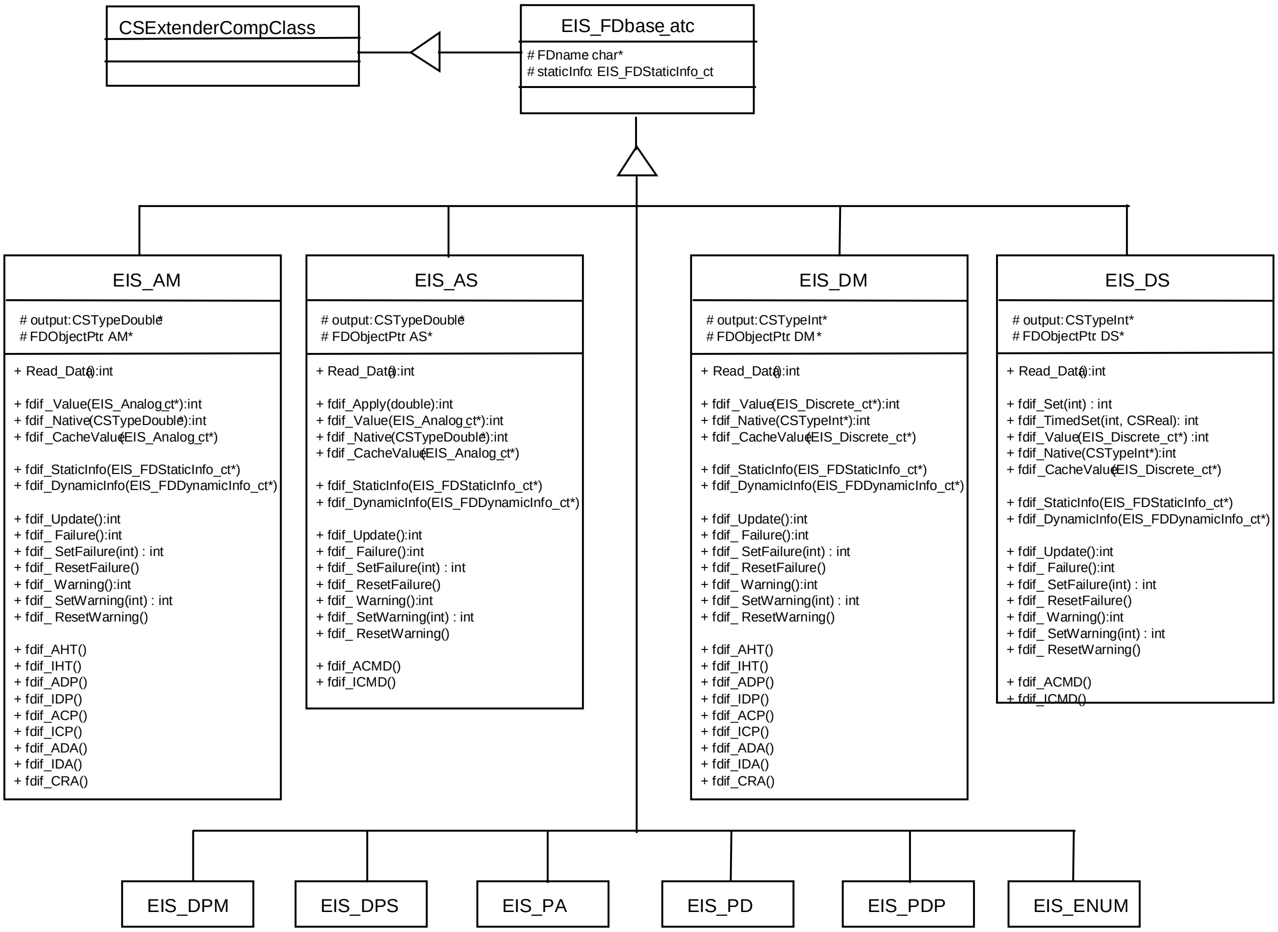
In this section list the development tools which will be used to implement the CSC. For example:

Tool	Use
ControlShell	Code development and user interface development
ProDev	Code Debugging SGI
Visual Toolkit	Code Debugging SUN
Perl	Enum .dat file generation (development)

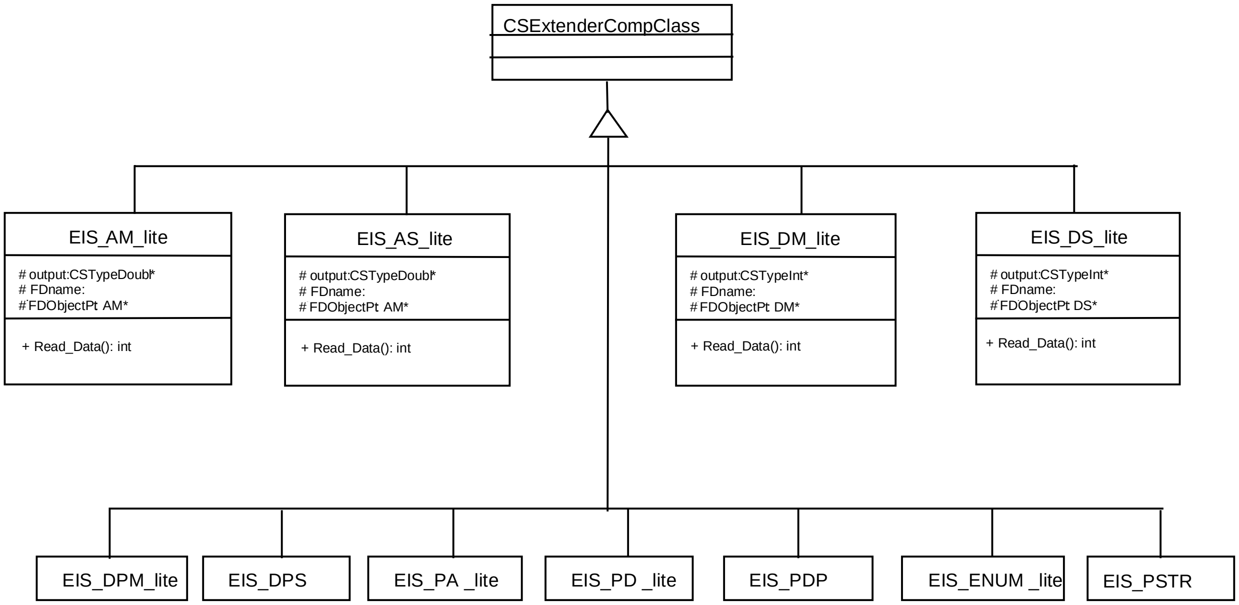
1.3.6 EIM Services External Interfaces

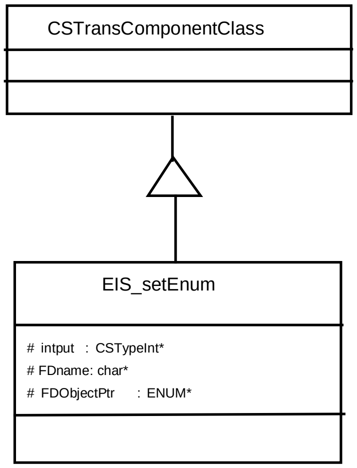
1.3.6.1 EIM Services Data Dictionary/Class Diagrams and Structures

1.3.6.1.1 FD Services Classes and Structures



Note: Duplicate methods cannot be moved up to base class because ControlShell does not allow inheritance of interface components (methods are in interface components). A future release of ControlShell will support interface inheritance, at which time the common methods will move to the base class.


CSExtenderCompClass
EIS_AM_lite
# output:CSTypeDoubl*
# FDname:
# FDObjectPt AM*
+ Read_Data(): int
EIS_AS_lite
# output:CSTypeDoubl*
# FDname:
# FDObjectPt AM*
+ Read_Data(): int
EIS_DM_lite
# output:CSTypeInt*
# FDname:
# FDObjectPt DM*
+ Read_Data(): int
EIS_DS_lite
# output:CSTypeInt*
# FDname:
# FDObjectPt DS*
+ Read_Data(): int
EIS_DPM_lite
EIS_DPS
EIS_PA _lite
EIS_PD _lite
EIS_PDP
EIS_ENUM _lite
EIS_PSTR


CSTransComponentClass
EIS_setEnum
# intput : CSTypeInt*
# FDname: char*
# FDObjectPtr : ENUM*

EIS_FDStaticInfo_ct			
	NAME:	TYPE:	DESCRIPTION:
	Name	char*	FD name
	Fdid	int	FD ID
	description	char*	FD description
	source	char*	Source of FD data
	units	char*	Engineering units of FD
	gateway	char*	Gateway processor ID
	RSYS	char*	TCID responsible system
	type	char*	FD data type
	subtype	char*	FD data subtype
	logical0state	char*	Logical zero state
	startBit	int	Starting bit position of FD data
	length	int	FD data length
	dataSetName	char*	Time homogenous data set name
	lowCount	int	Low count limit
	highCount	int	High count limit
	lowRange	CSReal	Low engineering value limit
	highRange	CSReal	High engineering value limit
	dbRSYS	char*	Databank responsible system

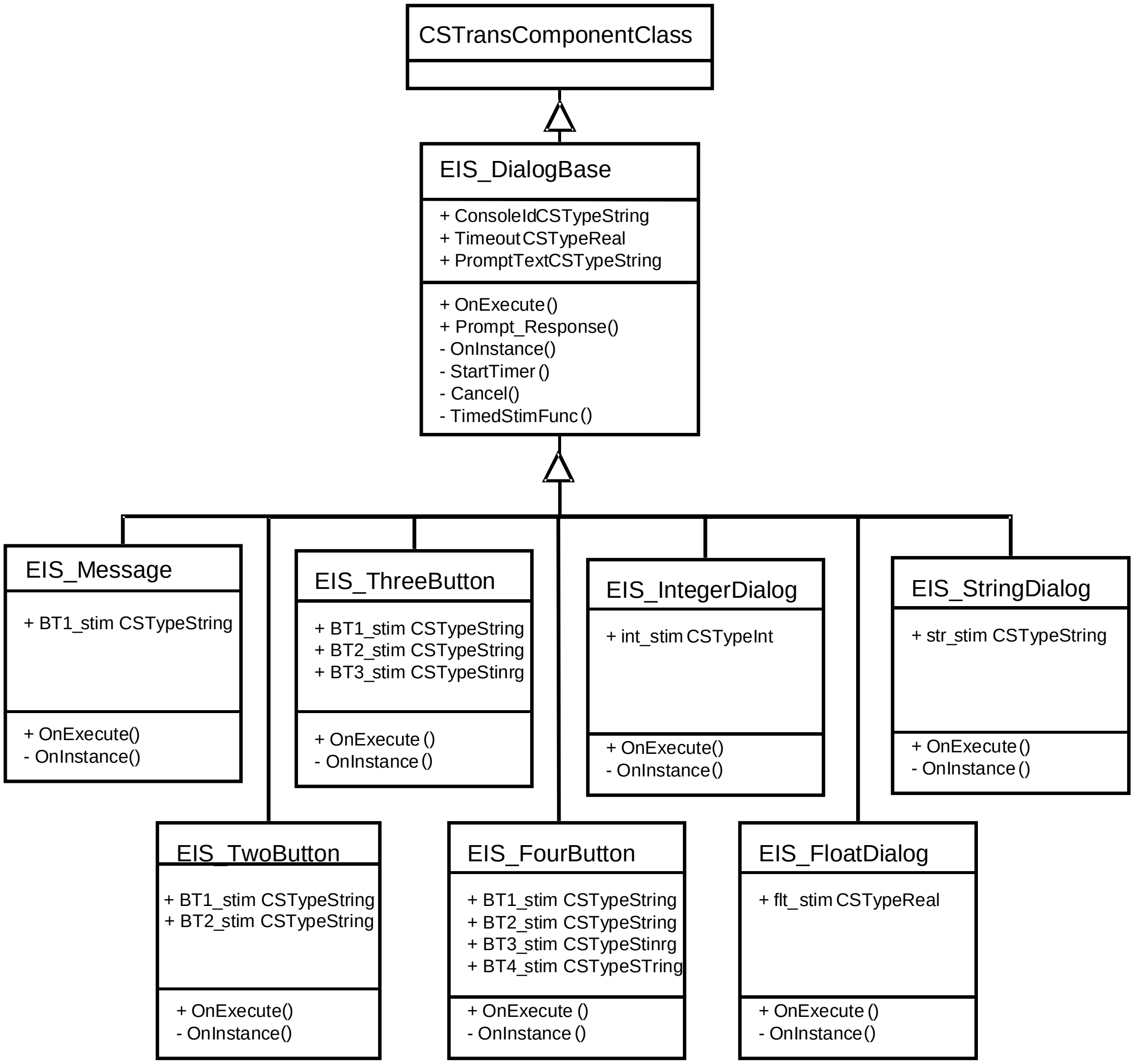
EIS_FDDynamicInfo_ct			
	NAME:	TYPE:	DESCRIPTION:
	healthTime	double	FD health update time
	valueTime	double	FD value update time
	failureCode	int	FD failure code
	warningCode	int	FD warning code

EIS_Analog_ct			
	NAME:	TYPE:	DESCRIPTION:
	value	CSReal	Analog value

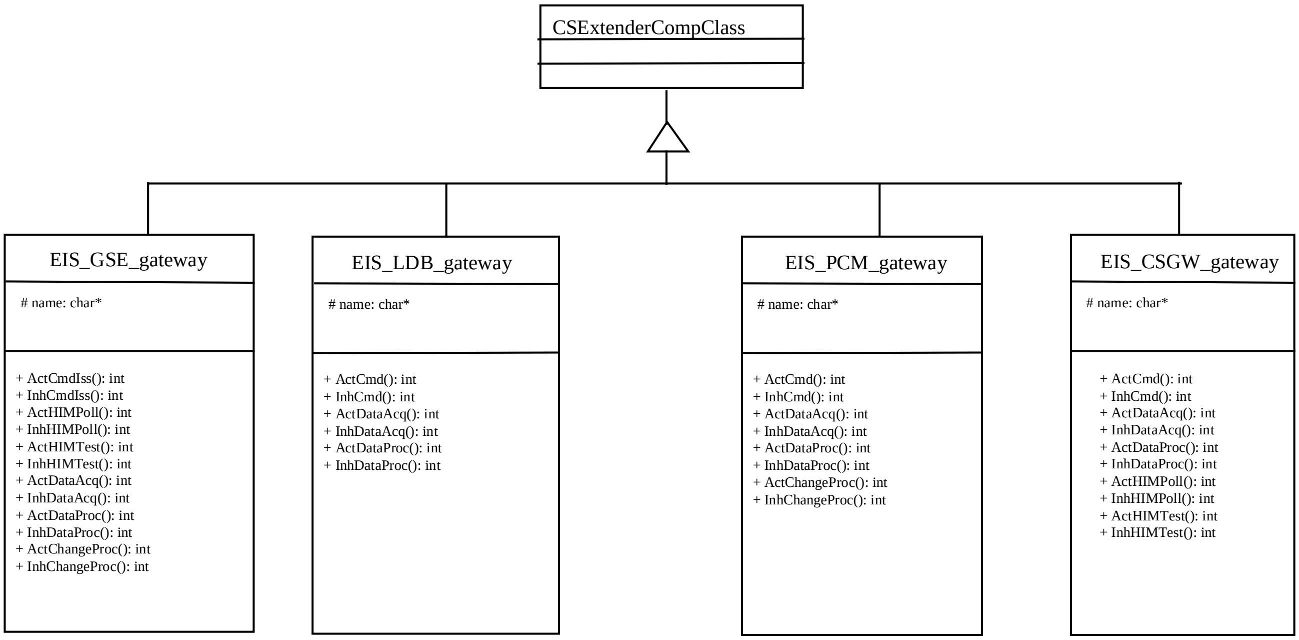
EIS_Discrete_ct			
	NAME:	TYPE:	DESCRIPTION:
	value	int	Discrete value

EIS_DigitalPattern_ct			
	NAME:	TYPE:	DESCRIPTION:
	value	int	Digital Pattern value

1.3.6.1.2 User Display Services Classes and Structures



1.3.6.1.3 Sub-System Services Classes



1.3.6.2 EIM Services Message Formats

1.3.6.2.1 FD Services Messages

Message Number = 296

Message Mnemonic = ASV_APPLICATIONS SWVS_FD_NOT_FOUND_EXCEPTION
Severity = Error

FD Services was unable to lookup FD in FDDictionary
FD Name: **I1**

Insert #1 = String – FD name

Help Information Context:

FD has not been Registered in FDDictionary or FD name is incorrectly spelled at ControlShell ™ ATC.

Message Number = 391

Message Mnemonic = ASV_APPLICATIONS SWVS_FDWRITE_EXCEPTION
Severity = Error

FD Services was unable to send FD object stimulus command set(), issue(), or apply().
FD Name: **I1**

Insert #1 = String – FD name

Help Information Context:

FD Services encountered an off-nominal condition with the Command Management interface.

Message Number = 392

Message Mnemonic = ASV_APPLICATIONS SWVS_FDREAD_EXCEPTION
Severity = Error

FD Services was unable to perform FD object method value(), native(), or update().
FD Name: **I1**

Insert #1 = String – FD name

Help Information Context:

FD Services encountered the FD health in an off-nominal state.

Message Number = 393

Message Mnemonic = ASV_APPLICATIONS SWVS_FDHEALTH_WRITE_EXCEPTION
Severity = Error

FD Services was unable to send FD object stimulus command set(), issue(), or apply().
FD Name: **I1**

Insert #1 = String – FD name

Help Information Context:

FD Services encountered the FD health in an off-nominal state.

Message Number = 397

Message Mnemonic = ASV_APPLICATIONS SWVS_FDTYPES_DON'T_MATCH_EXCEPTION
Severity = Error

FD Services was unable to lookup FD in FDDictionary because of type mismatch.
FD Name: **I1**

Insert #1 = String – FD name

Help Information Context:

Function Designator argument in FDDictionary lookup() method does not match type registered in FDDictionary.

1.3.6.3 EIM Services Display Formats

N/A

1.3.6.4 EIM Services Input Formats

N/A

1.3.6.5 Recorded Data

N/A

1.3.6.6 Local Storage Requirements and Formats

N/A

1.3.6.7 EIM Services Printer Formats

N/A

1.3.6.8 EIM Services File Formats

EIS provides a perl script that translates the TCID enum.data file into several <enumeration>.dat files, which are used by the applications developers. These <enumeration>.dat files give the application visibility to the enumerations contained in a TCID, in a manner useable by the ControlShell ™ drawing. Once the ControlShell ™ application developer includes the <enumeration>.dat file in the <application>_unix.dat file, the ControlShell ™ application can reference these enumeration literals as they would any global symbol.

The <enumeration>.dat files are located/stored in the $ASV_EIS_CSREP/include/enums subdirectory.

The following is the format of the TCID's enum.data file:

Field	Length
class number	6 chars
space	1 char
element value	20 chars
Space	1 char
element description	60 chars
space	1 char
enum class name	20 chars

The following is representative of a typical <enumeration>.dat file (for IND_CB1 enumeration):

```
CSType: rti_basic/CSTypeInt G_IND_CB1_CL
        BUILTIN/int val 1

CSType: rti_basic/CSTypeInt G_IND_CB1_OP
        BUILTIN/int val 2
```

The Constraint Management Services portion of EIS has some internal enumerations; these are also translated via perl script into <enumeration>.dat files, in order to allow ControlShell ™ application developers to access these enumerations like the TCID enumerations.

The following is the format of the eis_cms_enum.data file, also located in the $ASV_EIS_CSREP/include/enums subdirectory:

Field	Length
enum class name	variable
whitespace	variable
literal name	variable
whitespace	variable
literal value	variable
whitespace	variable
CS type	variable

1.3.6.9 Interprocess Communications

Interprocess communications will be done through CORBA. The following are the CORBA interfaces used by the UDS Prompting components:

```
module UDSDialog {
    interface EnumCallback
    {
        enum Response { ERR, BT1, BT2, BT3, BT4 };
        short  Prompt_Response( in Response resp );
    };

    interface StringCallback
    {
        short Prompt_Response( in string resp );
    };

    interface LongCallback
    {
        short Prompt_Response( in long resp );
    };

    interface FloatCallback
    {
        short Prompt_Response( in float resp );
    };

    interface Prompt
    {
        oneway void DisplayMessage(in string     Msg,
                            in string            Btn1_txt,
                            in EnumCallback      SeqCB );

        oneway void DisplayTwoButton(in string   Msg,
                            in string            Btn1_txt,
                            in string            Btn2_txt,
                            in EnumCallback      SeqCB );

        oneway void DisplayThreeButton(in string Msg,
                            in string            Btn1_txt,
                            in string            Btn2_txt,
                            in string            Btn3_txt,
                            in EnumCallback      SeqCB );

        oneway void DisplayFourButton(in string  Msg,
                            in string            Btn1_txt,
                            in string            Btn2_txt,
                            in string            Btn3_txt,
                            in string            Btn4_txt,
                            in EnumCallback      SeqCB );

        oneway void DisplayStringInput(in string Msg,
                            in string            default_val,
                            in StringCallback    SeqCB );

        oneway void DisplayLongInput(in string   Msg,
                            in long              default_val,
                            in LongCallback      SeqCB );

        oneway void DisplayFloatInput(in string  Msg,
                            in float             default_val,
                            in FloatCallback     SeqCB );


     };     // end of interface Prompt

};    // end of module UDSDialog
```

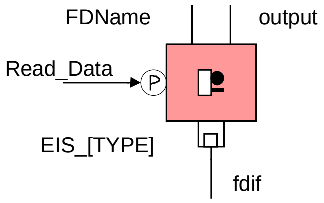
Note: CORBA interface still needs to be developed for IPC components if we are still tasked to develop them.

1.3.6.10 EIM Services External Interface Calls (e.g., API Calling Formats)

The following sections contain diagrams and tables which give basic descriptions of the ControlShell Components provided with this CSC.

1.3.6.10.1 FD Services Interfaces

1.3.6.10.1.1 Analog Measurement ATC



Based on EIS_[TYPE] Diagram

EIS_AM - use to obtain a specific FD’s CVT and OLDB information. Command FD’s gateway.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		Fdname	CSTypeString	Name of FD to utilize	
		output	CSTypeDouble	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.
	INTERFACES	METHOD:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
	fdif				
		Value	EIS_Analog_ct*	Passes FD’s current value by reference	Catch FDReadException and send system message #392. Return -1.
		Native	CSTypeDouble*	Passes FD’s current value by reference	Catch FDReadException and send system message #392. Return -1.
		CacheValue	EIS_Analog_ct*	Passes FD’s cached value by	

				reference	
		StaticInfo	EIS_FDStaticInfo_ct	Passes FD’s OLDB info. by reference	
		DynamicInfo	EIS_FDDynamicInfo_ct	Passes FD’s DynamicInfo by reference	
		Update	NoArg	Performs atomic read of FD data from CVT	Catch FDReadException and send system message #392. Return -1.
		Failure	NoArg	Returns FD’s failure health bit	
		SetFailure	int	Set FD’s failure health bit	
		ResetFailure	NoArg	Set FD’s failure health bit to zero	
		Warning	NoArg	Returns FD’s warning health bit	
		SetWarning	int	Set FD’s warning health bit	
		ResetWarning	NoArg	Set FD’s warning health bit to zero	
		AHT	NoArg	Activate HIM testing on FD	
		IHT	NoArg	Inhibit HIM testing on FD	
		ADP	NoArg	Activate data processing on FD	
		IDP	NoArg	Inhibit data processing on FD	
		ACP	NoArg	Activate change processing on FD	
		ICP	NoArg	Inhibit change processing on FD	
		ADA	NoArg	Activate data acquisition on FD	
		IDA	NoArg	Inhibit data acquisition on FD	
		CRA	int	Set sample rate of FD	

1.3.6.10.1.2 Analog Stimulus ATC

Based on EIS_[TYPE] Diagram

EIS_AS - use to obtain a specific FD’s CVT and OLDB information. Command FD and FD’s gateway.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeDouble	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.
	INTERFACES	METHOD:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
	fdif				
		Apply	double	Set FD’s value	Catch FDWriteException and FDHealthWriteException, send system message #393 or #391. Return –1.
		Value	EIS_Analog_ct*	Passes FD's current value by reference	Catch FDReadException and send system message #392. Return -1.
		Native	CSTypeDouble*	Passes FD's current value by reference	Catch FDReadException and send system message #392. Return -1.
		CacheValue	EIS_Analog_ct*	Passes FD's cached value by reference	
		StaticInfo	EIS_FDStaticInfo_ct	Passes FD's OLDB info. by reference	

		DynamicInfo	EIS_FDDynamicInfo_ct	Passes FD’s DynamicInfo by reference	
		Update	NoArg	Performs atomic read of FD data from CVT	Catch FDReadException and send system message #392. Return -1.
		Failure	NoArg	Returns FD’s failure health bit	
		SetFailure	int	Set FD’s failure health bit	
		ResetFailure	NoArg	Set FD’s failure health bit to zero	
		Warning	NoArg	Returns FD’s warning health bit	
		SetWarning	int	Set FD’s warning health bit	
		ResetWarning	NoArg	Set FD’s warning health bit to zero	
		ACMD	NoArg	Activate Cmd issuance on FD	
		ICMD	NoArg	Inhibit Cmd issuance on FD	

1.3.6.10.1.3 Discrete Measurement ATC

Based on EIS_[TYPE] Diagram

EIS_DM - use to obtain a specific FD's CVT and OLDB information. Command FD's gateway.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeInt	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.
	INTERFACES	METHOD:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
	fdif				
		Value	EIS_Discrete_ct*	Passes FD's current value by reference	Catch FDReadException and send system message #392. Return -1.
		Native	CSTypeInt*	Passes FD's current value by reference	Catch FDReadException and send system message #392. Return -1.
		CacheValue	EIS_Discrete_ct*	Passes FD's cached value by reference	
		StaticInfo	EIS_FDStaticInfo_ct	Passes FD's OLDB info. by reference	
		DynamicInfo	EIS_FDDynamicInfo_ct	Passes FD's DynamicInfo by reference	
		Update	NoArg	Performs atomic read of FD data from CVT	Catch FDReadException and send system message #392. Return -1.
		Failure	NoArg	Returns FD's failure health bit	
		SetFailure	int	Set FD's failure health bit	
		ResetFailure	NoArg	Set FD's failure health bit to zero	
		Warning	NoArg	Returns FD's warning health bit	
		SetWarning	int	Set FD's warning health bit	
		ResetWarning	NoArg	Set FD's warning health bit to zero	
		AHT	NoArg	Activate HIM testing on FD	
		IHT	NoArg	Inhibit HIM testing on FD	
		ADP	NoArg	Activate data processing on FD	
		IDP	NoArg	Inhibit data processing on FD	
		ACP	NoArg	Activate change processing on FD	
		ICP	NoArg	Inhibit change processing on FD	
		ADA	NoArg	Activate data acquisition on FD	
		IDA	NoArg	Inhibit data acquisition on FD	
		CRA	int	Set sample rate of FD	

1.3.6.10.1.4 Discrete Stimulus ATC

Based on EIS_[TYPE] Diagram

EIS_DS **-** use to obtain a specific FD’s CVT and OLDB information. Command FD and FD’s gateway.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeInt	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.
	INTERFACES	METHOD:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
	fdif				
		Set	int	Set FD’s value	Catch FDWriteException and FDHealthWriteException, send system message #393 or #391. Return –1.
		TimedSet	int , CSReal	Set FD's value for a specific time in seconds.	Catch FDWriteException and FDHealthWriteException, send system message #393 or #391. Return –1.
		Value	EIS_Discrete_ct*	Passes FD's current value by reference	Catch FDReadException and send system message #392. Return -1.
		Native	CSTypeInt*	Passes FD's current value by reference	Catch FDReadException and send system message #392. Return -1.
		CacheValue	EIS_Discrete_ct*	Passes FD's cached value by reference	
		StaticInfo	EIS_FDStaticInfo_ct	Passes FD's OLDB info. by reference	
		DynamicInfo	EIS_FDDynamicInfo_ct	Passes FD's DynamicInfo by reference	
		Update	NoArg	Performs atomic read of FD data from CVT	Catch FDReadException and send system message #392. Return -1.
		Failure	NoArg	Returns FD's failure health bit	
		SetFailure	int	Set FD's failure health bit	
		ResetFailure	NoArg	Set FD's failure health bit to zero	
		Warning	NoArg	Returns FD's warning health bit	
		SetWarning	int	Set FD's warning health bit	
		ResetWarning	NoArg	Set FD's warning health bit to zero	
		ACMD	NoArg	Activate Cmd issuance on FD	
		ICMD	NoArg	Inhibit Cmd issuance on FD	

1.3.6.10.1.5 Digital Pattern Measurement ATC

Based on EIS_[TYPE] Diagram

EIS_DPM - use to obtain a specific FD's CVT and OLDB information. Command FD's gateway.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeInt	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.
	INTERFACES	METHOD:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
	fdif				
		Value	EIS_DigitalPattern_ct*	Passes FD's current value by reference	Catch FDReadException and send system message #392. Return -1.
		Native	CSTypeInt*	Passes FD's current value by reference	Catch FDReadException and send system message #392. Return -1.
		CacheValue	EIS_DigitalPattern_ct*	Passes FD's cached value by reference	
		StaticInfo	EIS_FDStaticInfo_ct	Passes FD's OLDB info. by reference	
		DynamicInfo	EIS_FDDynamicInfo_ct	Passes FD's DynamicInfo by reference	
		Update	NoArg	Performs atomic read of FD data from CVT	Catch FDReadException and send system message #392. Return -1.
		Failure	NoArg	Returns FD's failure health bit	
		SetFailure	int	Set FD's failure health bit	
		ResetFailure	NoArg	Set FD's failure health bit to zero	
		Warning	NoArg	Returns FD's warning health bit	
		SetWarning	int	Set FD's warning health bit	
		ResetWarning	NoArg	Set FD's warning health bit to zero	
		AHT	NoArg	Activate HIM testing on FD	
		IHT	NoArg	Inhibit HIM testing on FD	
		ADP	NoArg	Activate data processing on FD	
		IDP	NoArg	Inhibit data processing on FD	
		ACP	NoArg	Activate change processing on FD	
		ICP	NoArg	Inhibit change processing on FD	
		ADA	NoArg	Activate data acquisition on FD	
		IDA	NoArg	Inhibit data acquisition on FD	
		CRA	int	Set sample rate of FD	

1.3.6.10.1.6 Digital Pattern Stimulus ATC

Based on EIS_[TYPE] Diagram

EIS_DPS **-** use to obtain a specific FD’s CVT and OLDB information. Command FD and FD’s gateway.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeInt	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.
	INTERFACES	METHOD:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
	fdif				
		Issue	int	Set FD’s value	Catch FDWriteException and FDHealthWriteException, send system message #393 or #391. Return –1.
		Value	EIS_DigitalPattern_ct*	Passes FD's current value by reference	Catch FDReadException and send system message #392. Return -1.
		Native	CSTypeInt*	Passes FD's current value by reference	Catch FDReadException and send system message #392. Return -1.
		CacheValue	EIS_DigitalPattern_ct*	Passes FD's cached value by reference	
		StaticInfo	EIS_FDStaticInfo_ct	Passes FD's OLDB info. by reference	
		DynamicInfo	EIS_FDDynamicInfo_ct	Passes FD's DynamicInfo by reference	
		Update	NoArg	Performs atomic read of FD data from CVT	Catch FDReadException and send system message #392. Return -1.
		Failure	NoArg	Returns FD's failure health bit	
		SetFailure	int	Set FD's failure health bit	
		ResetFailure	NoArg	Set FD's failure health bit to zero	
		Warning	NoArg	Returns FD's warning health bit	
		SetWarning	int	Set FD's warning health bit	
		ResetWarning	NoArg	Set FD's warning health bit to zero	
		ACMD	NoArg	Activate Cmd issuance on FD	
		ICMD	NoArg	Inhibit Cmd issuance on FD	

1.3.6.10.1.7 Psuedo Analog ATC

Based on EIS_[TYPE] Diagram

EIS_PA - use to obtain a specific FD’s CVT and OLDB information. Command FD.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeDouble	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.
	INTERFACES	METHOD:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
	fdif				
		Apply	double	Set FD’s value	Catch FDWriteException and FDHealthWriteException, send system message #393 or #391. Return –1.
		Value	EIS_Analog_ct*	Passes FD's current value by reference	Catch FDReadException and send system message #392. Return -1.
		Native	CSTypeDouble*	Passes FD's current value by reference	Catch FDReadException and send system message #392. Return -1.
		CacheValue	EIS_Analog_ct*	Passes FD's cached value by reference	
		StaticInfo	EIS_FDStaticInfo_ct	Passes FD's OLDB info. by reference	
		DynamicInfo	EIS_FDDynamicInfo_ct	Passes FD's DynamicInfo by reference	
		Update	NoArg	Performs atomic read of FD data from CVT	Catch FDReadException and send system message #392. Return -1.
		Failure	NoArg	Returns FD's failure health bit	
		SetFailure	int	Set FD's failure health bit	
		ResetFailure	NoArg	Set FD's failure health bit to zero	
		Warning	NoArg	Returns FD's warning health bit	
		SetWarning	int	Set FD's warning health bit	
		ResetWarning	NoArg	Set FD's warning health bit to zero	

1.3.6.10.1.8 *Psuedo Digital Pattern ATC*

Based on EIS_[TYPE] Diagram

EIS_PDP **-** use to obtain a specific FD’s CVT and OLDB information. Command FD.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeInt	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.
	INTERFACES	METHOD:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
	fdif				
		Issue	int	Set FD’s value	Catch FDWriteException and FDHealthWriteException, send system message #393 or #391. Return –1.
		Value	EIS_DigitalPattern_ct*	Passes FD's current value by reference	Catch FDReadException and send system message #392. Return -1.
		Native	CSTypeInt*	Passes FD's current value by reference	Catch FDReadException and send system message #392. Return -1.
		CacheValue	EIS_DigitalPattern_ct*	Passes FD's cached value by reference	
		StaticInfo	EIS_FDStaticInfo_ct	Passes FD's OLDB info. by reference	
		DynamicInfo	EIS_FDDynamicInfo_ct	Passes FD's DynamicInfo by reference	
		Update	NoArg	Performs atomic read of FD data from CVT	Catch FDReadException and send system message #392. Return -1.
		Failure	NoArg	Returns FD's failure health bit	
		SetFailure	int	Set FD's failure health bit	
		ResetFailure	NoArg	Set FD's failure health bit to zero	
		Warning	NoArg	Returns FD's warning health bit	
		SetWarning	int	Set FD's warning health bit	
		ResetWarning	NoArg	Set FD's warning health bit to zero	

1.3.6.10.1.9 Psuedo Discrete ATC

Based on EIS_[TYPE] Diagram

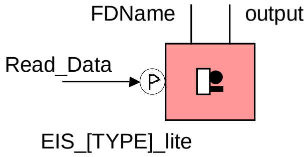
EIS_PD - use to obtain a specific FD’s CVT and OLDB information. Command FD.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeInt	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.
	INTERFACES	METHOD:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
	fdif				
		Set	int	Set FD’s value	Catch FDWriteException and FDHealthWriteException, send system message #393 or #391. Return –1.
		TimedSet	int , CSReal	Set FD's value for a specific time in seconds.	Catch FDWriteException and FDHealthWriteException, send system message #393 or #391. Return –1.
		Value	EIS_Discrete_ct*	Passes FD's current value by reference	Catch FDReadException and send system message #392. Return -1.
		Native	CSTypeInt*	Passes FD's current value by reference	Catch FDReadException and send system message #392. Return -1.
		CacheValue	EIS_Discrete_ct*	Passes FD's cached value by reference	
		StaticInfo	EIS_FDStaticInfo_ct	Passes FD's OLDB info. by reference	
		DynamicInfo	EIS_FDDynamicInfo_ct	Passes FD's DynamicInfo by reference	
		Update	NoArg	Performs atomic read of FD data from CVT	Catch FDReadException and send system message #392. Return -1.
		Failure	NoArg	Returns FD's failure health bit	
		SetFailure	int	Set FD's failure health bit	
		ResetFailure	NoArg	Set FD's failure health bit to zero	
		Warning	NoArg	Returns FD's warning health bit	
		SetWarning	int	Set FD's warning health bit	
		ResetWarning	NoArg	Set FD's warning health bit to zero	

1.3.6.10.1.10 Enumeration ATC

Based on EIS_[TYPE] Diagram

EIS_ENUM - use to obtain a specific FD's CVT and OLDB information. Command FD.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeInt	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.
	INTERFACES	METHOD:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
	fdif				
		Issue	int	Set FD's value	Catch FDWriteException and FDHealthWriteException, send system message #393 or #391. Return –1.
		Value	EIS_DigitalPattern_ct*	Passes FD's current value by reference	Catch FDReadException and send system message #392. Return -1.
		Native	CSTypeInt*	Passes FD's current value by reference	Catch FDReadException and send system message #392. Return -1.
		CacheValue	EIS_DigitalPattern_ct*	Passes FD's cached value by reference	
		StaticInfo	EIS_FDStaticInfo_ct	Passes FD's OLDB info. by reference	
		DynamicInfo	EIS_FDDynamicInfo_ct	Passes FD's DynamicInfo by reference	
		Update	NoArg	Performs atomic read of FD data from CVT	Catch FDReadException and send system message #392. Return -1.
		Failure	NoArg	Returns FD's failure health bit	
		SetFailure	int	Set FD's failure health bit	
		ResetFailure	NoArg	Set FD's failure health bit to zero	
		Warning	NoArg	Returns FD's warning health bit	
		SetWarning	int	Set FD's warning health bit	
		ResetWarning	NoArg	Set FD's warning health bit to zero	

1.3.6.10.1.11 Lightweight Analog Measurement ATC



EIS_AM_lite - use to obtain a specific FD's current value.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeDouble	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.

1.3.6.10.1.12 Lightweight Analog Stimulus ATC

Based on EIS_[TYPE]_lite diagram

EIS_AS_lite - use to obtain a specific FD's current value.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeDouble	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.

1.3.6.10.1.13 Lightweight Discrete Measurement ATC

Based on EIS_[TYPE]_lite diagram

EIS_DM_lite - use to obtain a specific FD’s current value.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeInt	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.

1.3.6.10.1.14 Lightweight Discrete Stimulus ATC

Based on EIS_[TYPE]_lite diagram

EIS_DS_lite - use to obtain a specific FD’s current value.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeInt	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.

1.3.6.10.1.15 Lightweight Digital Pattern Measurement ATC

Based on EIS_[TYPE]_lite diagram

EIS_DPM_lite - use to obtain a specific FD’s current value.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeInt	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.

1.3.6.10.1.16 Lightweight Digital Pattern Stimulus ATC

Based on EIS_[TYPE]_lite diagram

EIS_DPS_lite - use to obtain a specific FD's current value.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeInt	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.

1.3.6.10.1.17 Lightweight Psuedo Analog ATC

Based on EIS_[TYPE]_lite diagram

EIS_PA_lite - use to obtain a specific FD's current value.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeDouble	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.

1.3.6.10.1.18 Lightweight Psuedo Discrete ATC

Based on EIS_[TYPE]_lite diagram

EIS_PD_lite - use to obtain a specific FD's current value.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeInt	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.

1.3.6.10.1.19 Lightweight Psuedo Digital Pattern ATC

Based on EIS_[TYPE]_lite diagram

EIS_PDP_lite - use to obtain a specific FD’s current value.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeInt	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.

1.3.6.10.1.20 Lightweight Psuedo String FD ATC

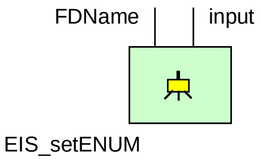
EIS_PSTR_lite - use to obtain a specific FD’s current value.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeInt	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.
		SetString	String	Set FD string value	Catch FDWriteException and FDHealthWriteException, send system message #393 or #391. Return –1.

1.3.6.10.1.21 Lightweight Enumeration ATC

Based on EIS_[TYPE]_lite diagram

EIS_ENUM_lite - use to obtain a specific FD's current value.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		output	CSTypeInt	FD current value. Read_Data() updates this value.	
	BUBBLES	NAME:	ARGUMENTS:	DESCRIPTION:	ERROR HANDLING:
		Read_Data	NoArg	Obtains current value of FD from CVT and stores it in output pin.	Catch FDReadException and send system message #392. Return -1.

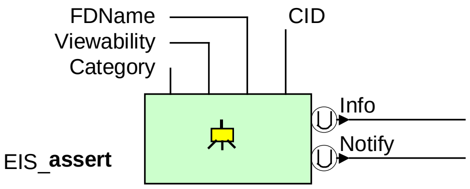
1.3.6.10.1.22 Set Enumeration STC



EIS_setENUM - use to set a specific enum FD's state.					
	PINS	NAME:	TYPE:	DESCRIPTION:	ERROR HANDLING:
		FDname	CSTypeString	Name of FD to utilize	
		input	CSTypeInt	FD enum state.	

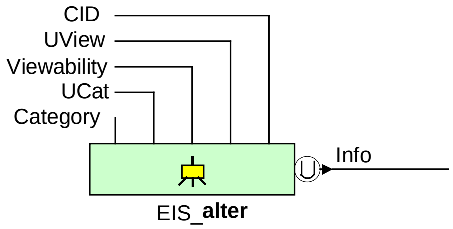
1.3.6.10.2 Constraint Management Services Interfaces

1.3.6.10.2.1 Assert STC



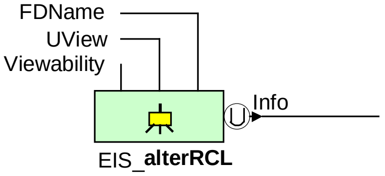
EIS_assert- STC used to assert a constraint					
	Pins	NAME	TYPE	CONNECTS TO:	USAGE: specifies-
		Category	CSTypeInt	G_EIS_CM_CAT_	Constraint Category
		Viewability	CSTypeInt	G_EIS_CM_VIEWABLE_	Constraint Viewability
		FDName	CSTypeString	Named String Signal	FD related to Constraint
		CID	CSTypeInt	Named Integer Signal (output)	Publishes the CID returned on assert
	Methods	NAME	TYPE	CONNECTS TO:	USAGE: specifies-
		Info	EIS_CInfo_cm	ATCs: EIS_AnON, EIS_DsON, EIS_DpON, EIS_EnON, EIS_OFF, EIS_PeriodTestingON, and EIS_SampleTestingON	Constraint Testing Algorithms
		Notify	EIS_Notify_cm	ATCs: EIS_SendStimByEvent	Real-Time Event Receiver
	Ret. Code	NAME	TYPE	CONNECTS TO:	MEANING

1.3.6.10.2.2 Alter STC



EIS_alter- STC used to alter a constraint					
	Pins	NAME	TYPE	CONNECTS TO:	USAGE: specifies-
		CID	CSTypeInt	Named Integer Signal	CID of Constraint to be Altered
		UCat	CSTypeInt	G_EIS_CM_UPDATE_	Whether or not to update Category
		Category	CSTypeInt	G_EIS_CM_CAT_	Constraint Category
		UView	CSTypeInt	G_EIS_CM_UPDATE_	Whether or not to update Viewability
		Viewability	CSTypeInt	G_EIS_CM_VIEWABLE_	Constraint Viewability
	Methods	NAME	TYPE	CONNECTS TO:	USAGE: specifies-
		Info	EIS_CInfo_cm	ATCS: EIS_AnON, EIS_DsON, EIS_DpON, EIS_EnON, EIS_OFF, EIS_PeriodTestingON, and EIS_SampleTestingON	Constraint Testing Algorithms
	Ret. Code	NAME	TYPE	CONNECTS TO:	MEANING

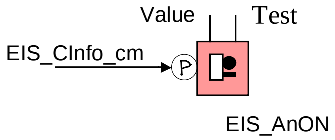
1.3.6.10.2.3 AlterRCL STC



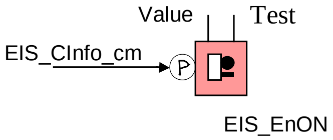
EIS_alter- STC used to alter a constraint					
	Pins	NAME	TYPE	CONNECTS TO:	USAGE: specifies-
		FDName	CSTypeString	Named String Signal	FD of RCL Constraint to be Altered
		UView	CSTypeInt	G_EIS_CM_UPDATE_	Whether or not to update Viewability
		Viewability	CSTypeInt	G_EIS_CM_VIEWABLE_	Constraint Viewability
	Methods	NAME	TYPE	CONNECTS TO:	USAGE: specifies-
		Info	EIS_CInfo_cm	ATCS: EIS_AnON, EIS_DsON, EIS_DpON, EIS_EnON, EIS_OFF, EIS_PeriodTestingON, and EIS_SampleTestingON	Constraint Testing Algorithms
	Ret. Code	NAME	TYPE	CONNECTS TO:	SPECIFIES

1.3.6.10.2.4 Info ATCs

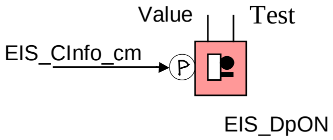
Analog Info ATC



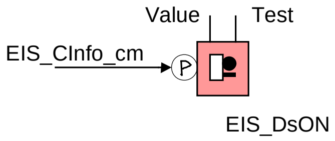
Enum Info ATC



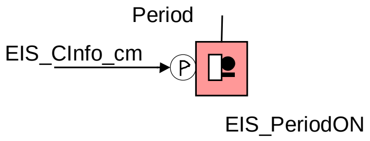
Digital Pattern Info ATC

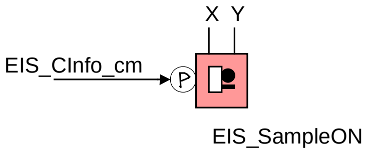


Discrete Info ATC



Boundary Info ATCs

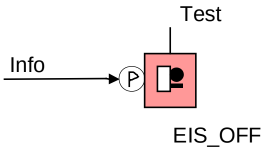




EIS_[TYPE]ON & EIS_[BoundaryTest]ON ATCs - used to specify constraint test algorithms on					
	Pins	NAME	TYPE	CONNECTS TO:	USAGE: specifies-
		Value	EIS_[UNIT]_ct	Named UnitTyped Signal	Value of the Test
		Test	EIS_Test_ct	G_EIS_TEST_	Test to be turned on
		Period	CSTypeInt	Named Int Signal	Period Value
		X,Y	CSTypeInt	Named Int Signals	X & Y of Sample Tests
	Methods	NAME	TYPE	CONNECTS TO:	USAGE:
		Info	EIS_CInfo_cm	STCS: EIS_assert, EIS_alterRCL, EIS_alter, EIS_activate, EIS_activateRCL, EIS_inhibit, EIS_inhibitRCL, and EIS_release	Connects test(s) to a Constraint

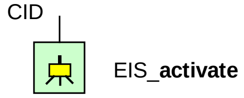
1.3.6.10.2.5 Explicit Off ATC

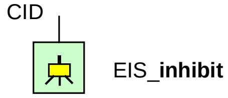
Explicit OFF ATC

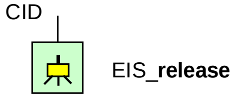


EIS_OFF- ATC used to explicitly specify a constraint test off					
	Pins	NAME	TYPE	CONNECTS TO:	USAGE: specifies-
		Test	EIS_Test_ct	G_EIS_TEST_	Test to be turned off
	Methods	NAME	TYPE	CONNECTS TO:	USAGE:
		Info	EIS_CInfo_cm	STCS: EIS_assert, EIS_alterRCL, EIS_alter, EIS_activate, EIS_activateRCL, EIS_inhibit, EIS_inhibitRCL, and EIS_release	Connects test(s) specification to a Constraint

1.3.6.10.2.6 Activate, Inhibit, and Release STCs

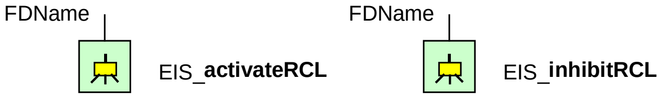






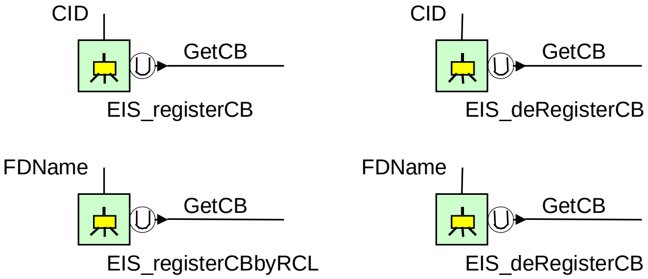
EIS_activate, EIS_inhibit, EIS_release STCs - used to provide their respecive API calls					
	Pins	NAME	TYPE	CONNECTS TO:	USAGE: specifies-
		CID	CSTypeInt	Named Int Signal	CID of Constraint to be activateds, inhibited, or released
	Ret. Code	NAME	TYPE	CONNECTS TO:	SPECIFIES

1.3.6.10.2.7 ActivateRCL and Inhibit RCL STCs



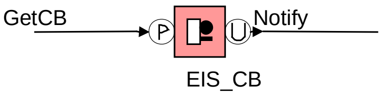
EIS_activateRCL, EIS_inhibitRCL, STCs - used to provide their respecive API calls for RCL constraints					
	Pins	NAME	TYPE	CONNECTS TO:	USAGE: specifies-
		FDName	CSTypeString	Named String Signal	FD of RCL Constraint to be activated or inhibited
	Ret. Code	NAME	TYPE	CONNECTS TO:	SPECIFIES

1.3.6.10.2.8 Callback STCs



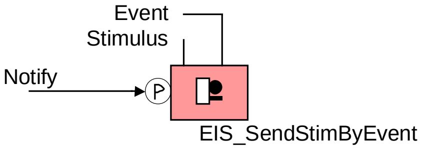
EIS_registerCB EIS_deregisterCB STCs - used to register a EIS_CB ATC for event notifications of a constraint					
	Pins	NAME	TYPE	CONNECTS TO:	USAGE: specifies-
		CID	CSTypeInt	Named Int Signal	CID of Constraint to listen for
		FDName	CSTypeString	Named String Signal	FDName of RCLConstraint to listen for
	Methods	NAME	TYPE	CONNECTS TO:	USAGE:
		GetCB	EIS_GetCB_ptr_cm	ATCS: EIS_CB	Gets the CB reference from EIS_CB
	Ret. Code	NAME	TYPE	CONNECTS TO:	SPECIFIES

1.3.6.10.2.9 Callback Component ATC



EIS_CB ATC - used to provide the Event Callback Corba Server to receive Real-Time Event Notifications					
	Methods	NAME	TYPE	CONNECTS TO:	USAGE:
		GetCB	EIS_GetCB_ptr_cm	STCS: EIS_registerCB, EIS_deregisterCB, EIS_registerCBbyRCL, & EIS_deregisterCBbyRCL	Provides the Event CB reference to the register/deregister CB STCs
		Notify	EIS_Notify_cm	ATCs: EIS_SendStimByEvent	Real-Time Event Receiver

1.3.6.10.2.10 Event Handler ATC



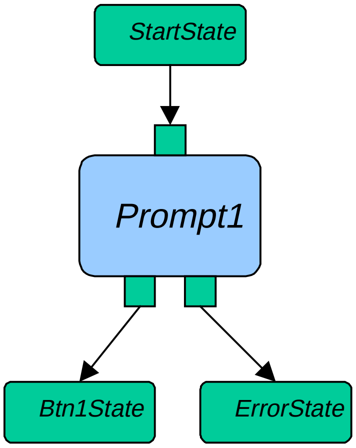
EIS_SendStimByEvent ATC - used to send a stimulus to an FSM in reaction to a Constraint Real-Time Event Notification					
	Pins	NAME	TYPE	CONNECTS TO:	USAGE: specifies-
		Event	EIS_CEvent_ct	G_EIS_CM_EVENT_	CID of Constraint to listen for
		Stimulus	Builtin/string	Primitive String Value	Name of the Stimulus to be sent
	Methods	NAME	TYPE	CONNECTS TO:	USAGE:
		Notify	EIS_Notify_cm	ATC: EIS_CB STC: EIS_assert	Real-Time Event Receiver

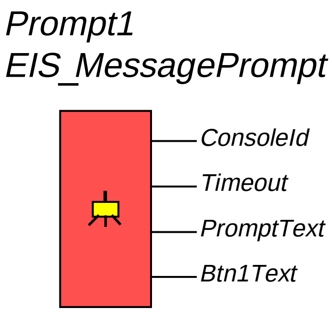
1.3.6.10.2.11 Compound and Summary Components

TBD- waiting on CMS Atlas DP3/APIs

1.3.6.10.3 User Display Services Interfaces

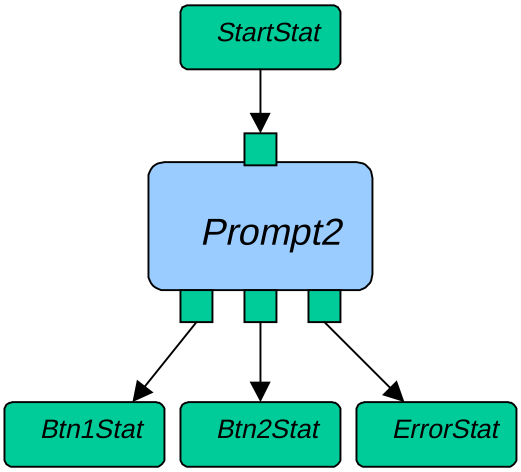
1.3.6.10.3.1 Message Prompt FSM

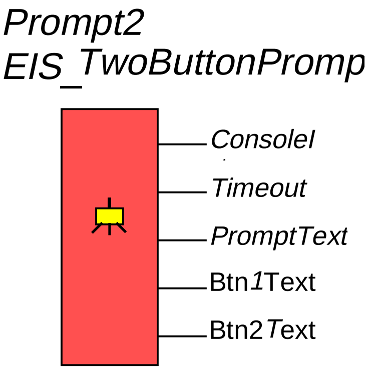




EIS_MessagePrompt- an FSM used to send a message to a user and receive an acknowledgement.					
	Pins	*NAME*	*TYPE*	*CONNECTS TO:*	*USAGE: specifies-*
		ConsoleId	*CSTypeString*	*Named String Signal*	*The name of the console server object that will display message Dialogs.*
		Timeout	*CSTypeInt*	*Named Integer Signal*	*The number of seconds for the FSM to wait for a user response before exiting through the error state. A zero means to never timeout.*
		PromptText	*CSTypeString*	*Named String Signal*	*The message that will be displayed in the Dialog box.*
		Btn1Text	*CSTypeString*	*Named String Signal*	*The text of the button the user clicks to acknowledge the message.*

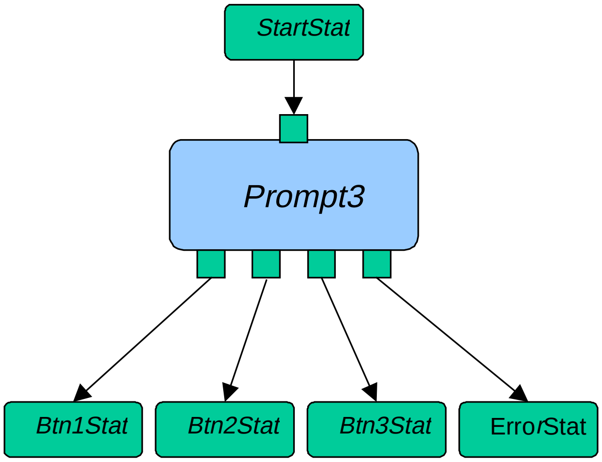
1.3.6.10.3.2 Two Button Prompt FSM

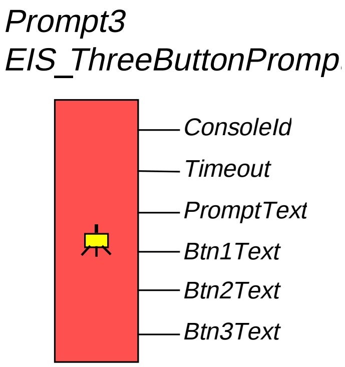




EIS_TwoButtonPrompt- an FSM used to query a user at a console and receive one of two possible responses.					
	Pins	*NAME*	*TYPE*	*CONNECTS TO:*	*USAGE: specifies-*
		ConsoleId	*CSTypeString*	*Named String Signal*	*The name of the console server object that will display message Dialogs.*
		Timeout	*CSTypeInt*	*Named Integer Signal*	*The number of seconds for the FSM to wait for a user response before exiting through the error state. A zero means to never timeout.*
		PromptText	*CSTypeString*	*Named String Signal*	*The message that will be displayed in the Dialog box.*
		Btn1Text	*CSTypeString*	*Named String Signal*	*The text of the first button displayed in the Dialog box.*
		Btn2Text	*CSTypeString*	*Named String Signal*	*The text of the second button displayed in the Dialog box.*

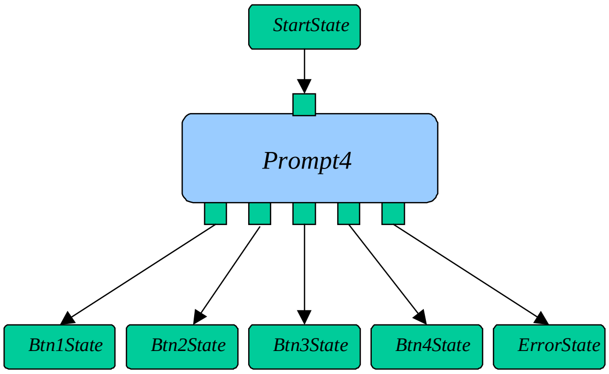
1.3.6.10.3.3 Three Button Prompt FSM

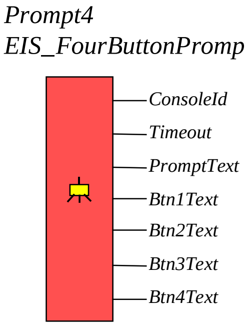




EIS_ThreeButtonPrompt- an FSM used to query a user at a console and receive one of three possible responses.					
	Pins	*NAME*	*TYPE*	*CONNECTS TO:*	*USAGE: specifies-*
		ConsoleId	*CSTypeString*	*Named String Signal*	*The name of the console server object that will display message Dialogs.*
		Timeout	*CSTypeInt*	*Named Integer Signal*	*The number of seconds for the FSM to wait for a user response before exiting through the error state. A zero means to never timeout.*
		PromptText	*CSTypeString*	*Named String Signal*	*The message that will be displayed in the Dialog box.*
		Btn1Text	*CSTypeString*	*Named String Signal*	*The text of the first button displayed in the Dialog box.*
		Btn2Text	*CSTypeString*	*Named String Signal*	*The text of the second button displayed in the Dialog box.*
		Btn3Text	*CSTypeString*	*Named String Signal*	*The text of the third button displayed in the Dialog box.*

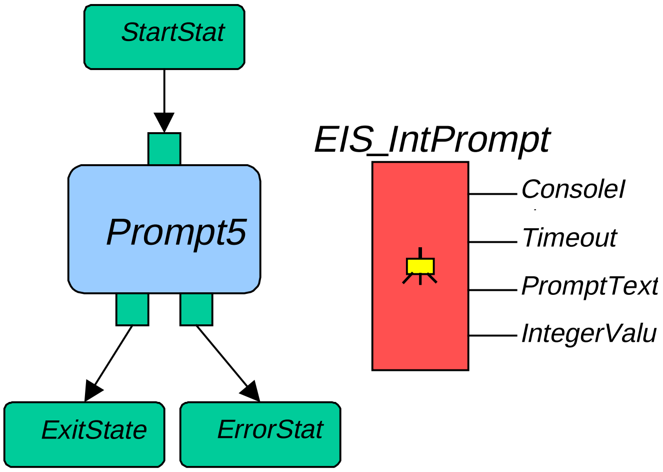
1.3.6.10.3.4 Four Button Prompt FSM





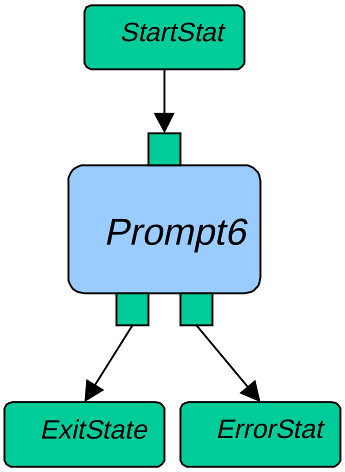
EIS_FourButtonPrompt- an FSM used to query a user at a console and receive one of four possible responses.					
	Pins	*NAME*	*TYPE*	*CONNECTS TO:*	*USAGE: specifies-*
		ConsoleId	*CSTypeString*	*Named String Signal*	*The name of the console server object that will display message Dialogs.*
		Timeout	*CSTypeInt*	*Named Integer Signal*	*The number of seconds for the FSM to wait for a user response before exiting through the error state. A zero means to never timeout.*
		PromptText	*CSTypeString*	*Named String Signal*	*The message that will be displayed in the Dialog box.*
		Btn1Text	*CSTypeString*	*Named String Signal*	*The text of the first button displayed in the Dialog box.*
		Btn2Text	*CSTypeString*	*Named String Signal*	*The text of the second button displayed in the Dialog box.*
		Btn3Text	*CSTypeString*	*Named String Signal*	*The text of the third button displayed in the Dialog box.*
		Btn4Text	*CSTypeString*	*Named String Signal*	*The text of the four button displayed in the Dialog box.*

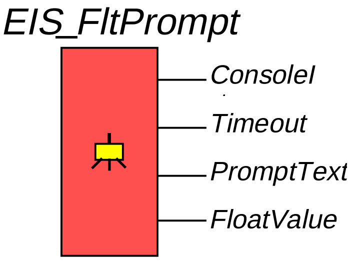
1.3.6.10.3.5 Integer Input Prompt FSM



EIS_IntPrompt- an FSM used to get an integer value from a user at a console.					
	Pins	*NAME*	*TYPE*	*CONNECTS TO:*	*USAGE: specifies-*
		ConsoleId	*CSTypeString*	*Named String Signal*	*The name of the console server object that will display message Dialogs.*
		Timeout	*CSTypeInt*	*Named Integer Signal*	*The number of seconds for the FSM to wait for a user response before exiting through the error state. A zero means to never timeout.*
		PromptText	*CSTypeString*	*Named String Signal*	*The message that will be displayed in the Dialog box.*
		IntegerValue	*CSTypeInt*	*Named Integer Signal*	*On entry into the fsm, it specifies the default value to be displayed in the input field. On exit, holds the value that the user entered.*

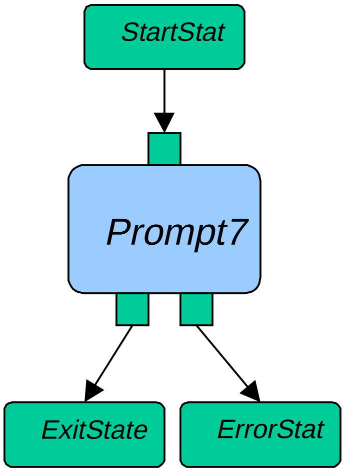
1.3.6.10.3.6 Floating Point Input Prompt FSM

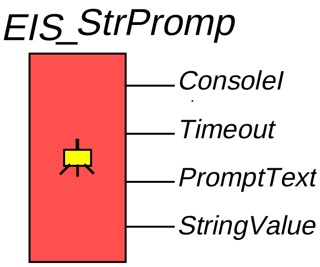




EIS_FltPrompt- an FSM used to get a floating point value from a user at a console.					
	Pins	*NAME*	*TYPE*	*CONNECTS TO:*	*USAGE: specifies-*
		ConsoleId	*CSTypeString*	*Named String Signal*	*The name of the console server object that will display message Dialogs.*
		Timeout	*CSTypeInt*	*Named Integer Signal*	*The number of seconds for the FSM to wait for a user response before exiting through the error state. A zero means to never timeout.*
		PromptText	*CSTypeString*	*Named String Signal*	*The message that will be displayed in the Dialog box.*
		FloatValue	*CSTypeDouble*	*Named Double Signal*	*On entry into the fsm, it specifies the default value to be displayed in the input field. On exit, holds the value that the user entered.*

1.3.6.10.3.7 String Input Prompt FSM

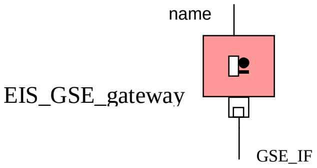




EIS_StrPrompt- an FSM used to get a string value from a user at a console.					
	Pins	*NAME*	*TYPE*	*CONNECTS TO:*	*USAGE: specifies-*
		ConsoleId	*CSTypeString*	*Named String Signal*	*The name of the console server object that will display message Dialogs.*
		Timeout	*CSTypeInt*	*Named Integer Signal*	*The number of seconds for the FSM to wait for a user response before exiting through the error state. A zero means to never timeout.*
		PromptText	*CSTypeString*	*Named String Signal*	*The message that will be displayed in the Dialog box.*
		StringValue	*CSTypeString*	*Named String Signal*	*On entry into the fsm, it specifies the default value to be displayed in the input field. On exit, holds the value that the user entered.*

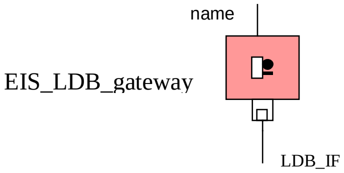
1.3.6.10.4 Sub-System Services Interfaces

1.3.6.10.4.1 *GSE Gateway ATC*



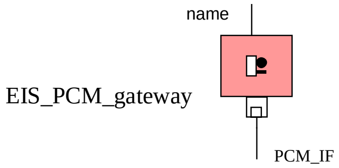
EIS_GSE_Gateway ATC - used to send commands to a GSE gateway					
	Pins	NAME	TYPE	CONNECTS TO:	USAGE: specifies-
		Name	CSTypeString	Named String Signal	Gateway name
	Interfaces	NAME:	METHOD	TYPE:	DESCRIPTION
		GSE_if		EIS_GSE_Gateway_if	Command Activate & Inhibit
			ActCmdIss	NoArg uses	
			InhCmdIss	NoArg uses	
			ActHIMPoll	NoArg uses	
			InhHIMPoll	NoArg uses	
			ActHIMTest	NoArg uses	
			InhHIMTest	NoArg uses	
			ActDataAcq	NoArg uses	
			InhDataAcq	NoArg uses	
			ActDataProc	NoArg uses	
			InhDataProc	NoArg uses	
			ActChangeProc	NoArg uses	
			InhChangeProc	NoArg uses	

1.3.6.10.4.2 *LDB Gateway ATC*



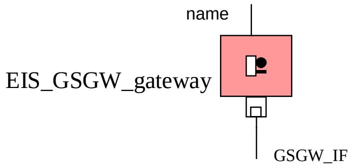
EIS_LDB_Gateway ATC - used to send commands to a LDB gateway					
	Pins	NAME	TYPE	CONNECTS TO:	USAGE: specifies-
		Name	CSTypeString	Named String Signal	Gateway name
	Interfaces	NAME:	METHOD	TYPE:	DESCRIPTION
		GSE_if		EIS_GSE_Gateway_if	
			ActCmd	NoArg uses	Command Activate
			InhCmd	NoArg uses	Command Inhibit
			ActDataAcq	NoArg uses	Activate Data Acquisition
			InhDataAcq	NoArg uses	
			ActDataProc	NoArg uses	
			InhDataProc	NoArg uses	

1.3.6.10.4.3 PCM Gateway ATC



EIS_PCM_Gateway ATC - used to send commands to a PCM gateway					
	Pins	NAME	TYPE	CONNECTS TO:	USAGE: specifies-
		name	CSTypeString	Named String Signal	Gateway name
	Interfaces	NAME:	METHOD	TYPE:	DESCRIPTION
		PCM_if		EIS_PCM_Gateway_if	
			ActCmd	NoArg uses	Command Activate
			InhCmd	NoArg uses	Command Inhibit
			ActDataAcq	NoArg uses	Activate Data Acquisition
			InhDataAcq	NoArg uses	Inhibit Data Acquisition
			ActDataProc	NoArg uses	Activate Data Processing
			InhDataProc	NoArg uses	Inhibit Data Processing
			ActChgProc	NoArg uses	Activate Change Processing
			InhChgProc	NoArg uses	Inhibit Change Processing
			GetStatus	NoArg uses	Get gateway status
			GetFDSubSys	NoArg uses	Get subsystem type of FD
			GetValue	NoArg uses	Get FD value from CVT

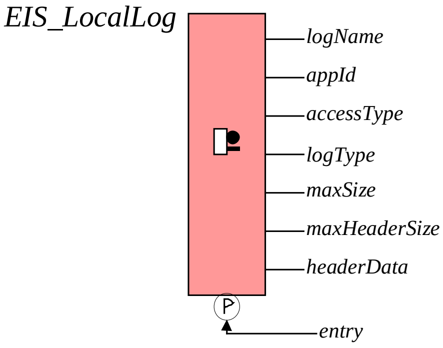
1.3.6.10.4.4 CSGW Gateway ATC



EIS_CSGW_Gateway ATC - used to send commands to a GSGW gateway					
	Pins	NAME	TYPE	CONNECTS TO:	USAGE: specifies-
		name	CSTypeString	Named String Signal	Gateway name
	Interfaces	NAME:	METHOD	TYPE:	DESCRIPTION
		CSGW_if		EIS_CSGW_Gateway_if	
			ActCmd	NoArg uses	Command Activate
			InhCmd	NoArg uses	Command Inhibit
			ActDataAcq	NoArg uses	Activate Data Acquisition
			InhDataAcq	NoArg uses	Inhibit Data Acquisition
			ActDataProc	NoArg uses	Activate Data Processing
			InhDataProc	NoArg uses	Inhibit Data Processing
			ActCmd	NoArg uses	Activate Command Issuance
			InhCmd	NoArg uses	Inhibit Command Issuance
			ActHimPoll	TwoIntArg uses	Activate HIM Polling
			InhHimPoll	TwoIntArg uses	Inhibit HIM Polling
			ActHimTest	IntArg uses	Activate HIM testing
			InhHimTest	IntArg uses	Inhibit HIM testing

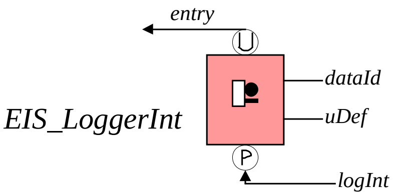
1.3.6.10.5 Local Logging Services Interfaces

1.3.6.10.5.1 Local Logger ATC



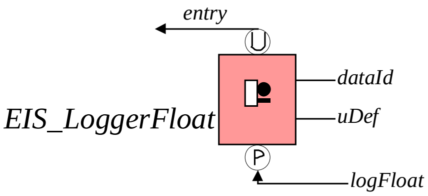
EIS_LocalLog- an ATC used to create a Local Log.					
	Pins	*NAME*	*TYPE*	*CONNECTS TO:*	*USAGE: specifies-*
		logName	*CSTypeString*	*Named String Signal*	*Specifies the full pathname of the local log file.*
		appId	*CSTypeString*	*Named String Signal*	*A unique Identifier of the application that will be using the log.*
		accessType	*CSTypeInt*	*G_LLS_CREATE,* *G_LLS_APPEND,* *G_LLS_READ*	*Type of file access.*
		logType	*CSTypeInt*	*G_LLS_FLAT_SINGLE,* *G_LLS_FLAT_MULTI,* *G_LLS_CIRCULAR*	*Type of log file to create.*
		maxSize	*CSTypeInt*	*Named Integer Signal*	*Maximum size of log in bytes*
		maxHeaderSize	*CSTypeInt*	*Named Integer Signal*	*Maximum size of header in bytes*
		headerData	*CSTypeString*	*Named String Signal*	*Header to go into log file.*
	Methods	*NAME*	*TYPE*	*CONNECTS TO:*	*USAGE: specifies-*
	Provides	*entry*	*EIS_Logging_cm*	*ATCs:* *EIS_LoggerInt,* *EIS_LoggerFlt,* *EIS_LoggerStr,* *EIS_LoggerAnalog,* *EIS_LoggerDescrete,* *EIS_LoggerDigitalPattern*	*A data record to be logged.*

1.3.6.10.5.2 Integer Logger ATC



EIS_LoggerInt- an ATC used to send integer values as records to a Local Log or the Local System Log.					
	Pins	*NAME*	*TYPE*	*CONNECTS TO:*	*USAGE: specifies-*
		dataId	*CSTypeString*	*Named String Signal*	*Used to identify the data being logged.*
		uDef	*CSTypeInt*	*Named Integer Signal*	*A user defined field used during delogging to select the record it identifies.*
	Methods	*NAME*	*TYPE*	*CONNECTS TO:*	*USAGE: specifies-*
	Uses	*entry*	*EIS_Logging_cm*	*ATCs:* *EIS_LocalLog,* *EIS_SystemLog*	*Connects to a specific log component. Sends a data record to the log.*
	Provides	*logInt*	*IntArg_cm*	*Any component that uses an IntArg_cm bubble method.*	*Called to log a record to the connected log.*

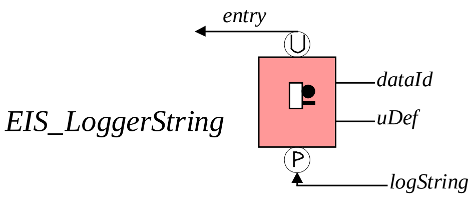
1.3.6.10.5.3 Float Logger ATC



EIS_LoggerFloat- an ATC used to send floating point values as records to a Local Log or the Local System Log.					
	Pins	*NAME*	*TYPE*	*CONNECTS TO:*	*USAGE: specifies-*
		dataId	*CSTypeString*	*Named String Signal*	*Used to identify the data being logged.*
		uDef	*CSTypeInt*	*Named Integer Signal*	*A user defined field used during delogging to select the record it identifies.*
	Methods	*NAME*	*TYPE*	*CONNECTS TO:*	*USAGE: specifies-*
	Uses	*entry*	*EIS_Logging_cm*	*ATCs:* *EIS_LocalLog,* *EIS_SystemLog*	*Connects to a specific log component. Sends a data record to the log.*
	Provides	*logFloat*	*DoubleArg_cm**	*Any component that uses a DoubleArg_cm bubble method.*	*Called to log a record to the connected log.*

**There is no DoubleArg_cm CS method in the cs_core repository. One will have to be created there or in some other repository.*

1.3.6.10.5.4 String Logger ATC



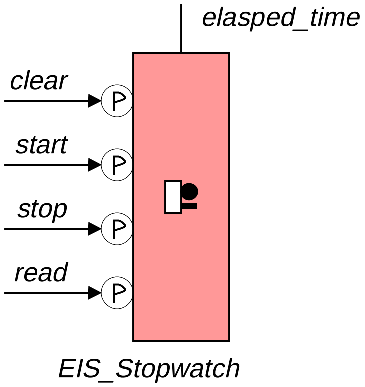
EIS_LoggerString- an ATC used to send strings as records to a Local Log or the Local System Log.					
	Pins	*NAME*	*TYPE*	*CONNECTS TO:*	*USAGE: specifies-*
		dataId	*CSTypeString*	*Named String Signal*	*Used to identify the data being logged.*
		uDef	*CSTypeInt*	*Named Integer Signal*	*A user defined field used during delogging to select the record it identifies.*
	Methods	*NAME*	*TYPE*	*CONNECTS TO:*	*USAGE: specifies-*
	Uses	*entry*	*EIS_Logging_cm*	*ATCs:* *EIS_LocalLog,* *EIS_SystemLog*	*Connects to a specific log component. Sends a data record to the log.*
	Provides	*logString*	*StringArg_cm**	*Any component that uses a StringArg_cm bubble method.*	*Called to log a record to the connected log.*

**There is no StringArg_cm CS method in the cs_core repository. One will have to be created there or in some other repository.*

Note: Other, more specific logger components can be added as needed. Good examples of this might be an EIS_LoggerAnalog, an EIS_LoggerDiscrete, and an EIS_LoggerDigitalPattern.

1.3.6.10.6 Timer Services Interfaces

1.3.6.10.6.1 TS Stopwatch ATC



TS_Stopwatch ATC - *used for stopwatch capability*					
	Pins	*NAME*	*TYPE*	*CONNECTS TO:*	*USAGE: specifies-*
		elapsed_time	*CSTypeInt*	*NamedIntSignal*	*Current value of stopwatch.*
	Bubbles	*NAME:*	*ARGUMENTS*	*DESCRIPTION:*	
		clear	*NoArg provides*	*Clear value of stopwatch.*	
		start	*NoArg provides*	*Start the stopwatch.*	
		stop	*NoArg provides*	*Stop the stopwatch.*	
		read	*NoArg provides*	*Read the stopwatch.*	

1.3.6.11 EIM Services Table Formats

N/A

1.3.7 EIM Services Test Plan

1.3.7.1 FD Services

FD Service Atomic Component
Test Objective: Test the following EIS capabilities to issue commands to FD’s:

- EIM Services shall provide the capability for Applications SW to read a FD’s value.
- EIM Services shall provide the capability for Applications SW to read a FD’s native value.
- EIM Services shall provide the capability for Applications SW to read a FD’s cache value.
- EIM Services shall provide the capability for Applications SW to read a FD’s static information (OLDB).
- EIM Services shall provide the capability for Applications SW to read a FD’s dynamic information.
- EIM Services shall provide the capability for Applications SW to perform an update of a FD.
- EIM Services shall provide the capability for Applications SW to read a FD’s failure bit.
- EIM Services shall provide the capability for Applications SW to read a FD’s warning bit.
- EIM Services shall provide the capability for Applications SW to set a FD’s failure bit.
- EIM Services shall provide the capability for Applications SW to set a FD’s warning bit.

- EIM Services shall provide the capability for Applications SW to set a FD’s value.
- EIM Services shall provide the capability for Applications SW to inhibit data acquisition on a FD.
- EIM Services shall provide the capability for Applications SW to activate data acquisition on a FD.
- EIM Services shall provide the capability for Applications SW to activate or inhibit data processing on a FD.
- EIM Services shall provide the capability for Applications SW to activate or inhibit change processing on a FD.
- EIM Services shall provide the capability for Applications SW to set sample rate of FD.
- EIM Services shall provide the capability for Applications SW to activate or inhibit HIM testing on a FD.

Test Approach Summary: The test conductor will run a ControlShell application which issues one or more of each of the FD commands. Output from the ControlShell application will be examined to make certain that the correct FD Services API calls occur.

Test Platform/Dependencies:

- CCP
  Sub-system services API stub

1.3.7.2 Constraint Management

1.3.7.2.1 Test Case 1 – Atomic Constraint API

Test Objective: Test the following EIS capabilities to issue commands to Constraint Management:

- EIM Services shall provide a capability for Applications SW to Assert, Alter, Activate, Inhibit, and Release Atomic Constraints.
- EIM Services will provide a capability to Alter, Activate, and Inhibit RCL Constraints by FDName.
- The EIM Interface to the Constraint Management API will be through EIS Constraint STCs and ATCs.

Test Approach Summary: The test conductor will run a ControlShell application which assert, alter, activate, and inhibit multiple constraints on multiple FDs of different types. The test will demonstrate the use of constraints in the EIM environment. Output from the ControlShell application will be examined to make certain that the correct actions occur.

Test Platform/Dependencies:

- CCP- Constraint Test Case EIM Application running
- DDP- Fully initialized with Constraint Manager running or just Constraint Manager Stub
- CCWS- Test Case Client Application and Constraint Viewer (if available)

1.3.7.2.2 Test Case 2 – Compound and Summary Constraint API

Test Objective: Test the following EIS capabilities to issue commands to Constraint Management:

- EIM Services shall provide a capability for Applications SW to Assert, Alter, Activate, Inhibit, and Release Compound and Summary Constraint.
- The EIM Interface to the Constraint Management API will be through EIS Constraint STCs and ATCs.

Test Approach Summary: The test conductor will run a ControlShell application which assert, alter, activate, and inhibit atomic, compound, and summary constraints on multiple FDs of different types. The test will demonstrate the use of compound and summary constraints in the EIM environment. Output from the ControlShell application will be examined to make certain that the correct actions occur.

Test Platform/Dependencies:

- CCP- Constraint Test Case EIM Application running
- DDP- Fully initialized with Constraint Manager running or just Constraint Manager Stub
- CCWS- Test Case Client Application and Constraint Viewer (if available)

1.3.7.2.3 Test Case 3- Constraint Event Handling

Test Objective: Test the following EIS capabilities to handle Constraint Notifications:

- EIM Services shall provide a capability to receive and react to Real-time Constraint Notifications from Constraint Management.
- Generic Event Handling ATC components will be provided to handle Constraint Notifications and send Stimuli to EIM FSMs.

Test Approach Summary: The test conductor will run a ControlShell application which assert some constraints and cause those constraints to send event notifications and send a stimulus to an FSM. Also, the test will demonstrate some of the capabilities required for RCL implementation. Output from the ControlShell application will be examined to make certain that the correct actions occur.

Test Platform/Dependencies:

- CCP- Constraint Test Case EIM Application running
- DDP- Fully initialized with Constraint Manager running
- CCWS- Test Case Client Application and Constraint Viewer (if available)

1.3.7.3 User Display Services

1.3.7.4 Sub-System Services

GSE Gateway Atomic Component

Test Objective: Test the following EIS capabilities to issue commands to GSE gateways:

- EIM Services shall provide the capability for Applications SW to inhibit data acquisition on a gateway.
- EIM Services shall provide the capability for Applications SW to activate data acquisition on a GSE gateway.
- EIM Services shall provide the capability for Applications SW to activate or inhibit data processing on a gateway.
- EIM Services shall provide the capability for Applications SW to activate or inhibit change processing on a gateway.
- EIM Services shall provide the capability for Applications SW to activate or inhibit command issuance on a GSE gateway.
- EIM Services shall provide the capability for Applications SW to activate or inhibit HIM polling on a GSE gateway.
- EIM Services shall provide the capability for Applications SW to activate or inhibit HIM testing on a GSE gateway.

Test Approach Summary: The test conductor will run a ControlShell application which issues one or more of each of the GSE gateway commands. Output from the ControlShell application will be examined to make certain that the correct Sub-system Services API calls occur.

Test Platform/Dependencies:

- CCP
- Sub-system services API stub

1.3.8 Issues

1. What are the requirements of the IPC components? Should be resolved in meeting scheduled for 6/19.
2. Is there an enumeration value/integer associated with the ENUM_ERROR literal?